“十五”国家重点电子出版物规划项目·计算机知识普及和软件开发系列

编程宝典
2002
4

北京希望电子出版社 总策划
夏云庆 编 写

Visual C++ 6.0 数据库高级编程

本书用7个典型范例详细介绍了用VISUAL C++6.0进行数据库开发的方法和技巧，且为每个编程范例提供了详细的操作步骤和源程序

中国科学出版集团
北京希望电子出版社

“十五”国家重点电子出版物规划项目·计算机知识普及和软件开发系列

编程宝典
2002
4

北京希望电子出版社 总策划
夏云庆 编 写

Visual C++ 6.0 数据库高级编程

本书用7个典型范例详细介绍了用VISUAL C++6.0进行数据库开发的方法和技巧，且为每个编程范例提供了详细的操作步骤和源程序

中国科学出版集团
北京希望电子出版社

内 容 简 介

本书以实例详解的方式，循序渐进地介绍了如何用 Visual C++ 6.0 开发数据库的方法和技巧。

本书内容共 3 篇和 3 个附录组成，由 10 章组成，第 1 篇（基础篇）的主要内容有数据库原理与访问、COM 与数据库访问、数据库开发过程、VC++数据库开发基础；第 2 篇（实例篇）通过 6 个编程实例详细介绍了 ODBC，DAO，OLE DB 以及 ADO 的客户端数据库的访问技术；第 3 篇（高级篇）通过 2 个实例介绍了 OLE DB 和 ADO 的高级编程技术。附录 A 是数据库访问的错误代码；附录 B 介绍了数据库编程资源网站；附录 C 是配套光盘内容结构图。

本书为所有编程实例提供了详细的操作步骤和源程序，对于用 Visual C++6.0 进行数据库应用开发人员具有很强的指导价值，也可以作为计算机软件专业本科生、硕士研究生的试验辅导教材。

光盘内容：1. 本教程中编程实例的源代码;2. 本教程中数据库应用中涉及的数据库文件。

系 列 盘 书：“十五”国家重点电子出版物规划项目 计算机知识普及和软件开发系列
编程宝典 2002 （4）

盘 书 名：Visual C++ 6.0 数据库高级编程

总 策 划：北京希望电子出版社

文本著作者：夏云庆 编写

C D 制作者：希望多媒体开发中心

C D 测试者：希望多媒体测试部

责任编辑：周 艳

出版、发行者：北京希望电子出版社

地 址：北京中关村大街 26 号，100080

网址: www.bhp.com.cn E-mail: lwm@hope.com.cn

电话: 010-62562329, 62541992, 62637101, 62637102, 62633308, 62633309
（图书发行和技术支持）
010-62613322-215（门市） 010-62547735（编辑部）

经 销：各地新华书店、软件连锁店

排 版：希望图书输出中心 董淑红

C D 生产者：北京中新联光盘有限责任公司

文本印刷者：北京双青印刷厂

开本 / 规格：787 毫米×1092 毫米 16 开本 21.5 印张 503.88 千字

版次 / 印次：2002 年 1 月第 1 版 2002 年 4 月第 2 次印刷

本 版 号：ISBN 7-900088- 07 -5

定 价：35.00 元（本版 CD）

编 者 的 话

信息技术是21世纪最有发展潜力的技术之一，信息的表示、获取、存储以及利用导致了新世纪软件技术的不断发展，数据库技术是所有信息技术的基础，离开了数据，信息便没有了立足之地，因而显示出数据库技术的关键地位。

数据库应用主要是面向各种信息的，包括商业的和非商业的，包括元数据和多媒体信息。当大量数据涌进数据库的时候，数据库的访问技术就成了瓶颈，同时也激励了数据库访问技术的不断发展。

VC++是众多编程语言中最突出的一种，无论底层的还是高层的操作接口，它都能够使用，而不受到开发需求的限制，因此说VC++对数据库访问技术的支持是最彻底的。从ODBC API到MFC的ODBC类，从DAO到OLE DB，VC++可以采用的数据库访问技术十分广泛。

本书对这些技术不做绝对的评价，因为每一种数据库访问技术都有其存在的价值，都有其它技术不能比拟的优势。评价一个技术的唯一标准是，它是不是适用于具体的应用需求。本书以实用为目的，结合具体的数据库应用，选择了7个最有代表性的实例，分别介绍ODBC API、MFC的ODBC类、DAO、OLE DB以及ADO的客户数据库访问技术和应用开发过程。本书还通过ADO数据库组件开发实例和OLE DB服务器程序的开发实例展开了数据库应用开发的高级话题。

本书是作者多年从事数据库开发实践的一些经验总结，是程序员对数据库访问技术灵活把握的很有价值的参考资料。本书所附的光盘里含有全部实例的源代码；在这里作者愿意毫无保留地将它们赠送给读者。

作 者

ANJS 01/01

~~ANJS 03/05~~

目　录

第 1 篇　基础篇

第 2 篇　实例篇

第 3 篇 高级话题

第1篇 基础篇

数据库技术作为计算机应用领域的重要组成部分，已经渗透到社会生活的方方面面。小到基本的公司日常管理，大到互联网的电子商务，都刻着数据库的印记。电子时代的到来，使数据库技术逐渐走进每个人的生活。数据库技术，小到基本的桌面应用，大到企业级的大型事务处理，它已经无所不在，无所不及。数据库技术为软件行业带来了巨大的生机和活力，数据库编程已经成为世界软件生产的重要内容。

为了使您快速了解数据库技术，掌握 VC++数据库编程的基本知识，本篇介绍如下内容：

- 第 1 章 数据库原理与访问
- 第 2 章 COM 与数据库访问
- 第 3 章 数据库开发过程
- 第 4 章 VC++数据库开发基础

第 1 章　数据库原理与访问

1.1　数据库基本原理

1.1.1　概述

数据库技术的发展历程

数据库是现代计算机应用的一个重要组成部分，是人们有效地进行数据存储、共享和处理的工具。

数据库技术的发展经过了 40 多年的历程。1963 年，C.W.Bachman 设计开发的 IDS（Integrated Data Store）系统开始投入运行，使多个 COBOL 程序可以共享数据库。1968 年，网状数据库系统 TOTAL 出现。1969 年，McGee 开发层次式数据库系统，发布了 IBM 的 IMS 系统。1970 年，IBM 公司 San Jose 研究所的 E.F.Code 发表了题为“大型共享数据库数据的关系模型”的著名论文，树立了关系型数据库的新的里程碑，E.F.Code 因此获得 1981 年度的 ACM 图灵奖，IBM San Jose 研究所也在 1976 年研制出在 IBM 370 机器上运行的 SYSTEM R 关系型数据库管理系统。1979 年，ORACLE 公司推出了第一个商品化的关系型数据库系统 ORACLE 2.0。80 年代至今，是数据库技术发展的成熟时期，这个时期出现了众多的大型数据库系统，包括 IBM 的 DB2、微软的 SQL Server、Sybase 以及 Informix 相继出现，使数据库系统呈现出夺目的光彩。

数据库系统的优势

数据库同文件相比，有以下优势：

首先，数据库中的数据是高度结构化的，不仅考虑数据项之间的关系，还考虑了记录类型之间的关系，从而反映出现实中的信息实体。

其次，数据库中的数据是面向系统而不是面向应用的，因此数据库的数据比文件系统的共享程度高，面向系统的另一个好处是信息结构稳定，易于扩展。

第三，数据库系统比文件系统有更高的独立性。为了实现这种独立性，数据库系统往往拥有比特定应用更多的数据，对于特定的应用只提供局部的逻辑结构，保持应用的逻辑独立性。

第四，数据库系统具有较好的数据安全性和一致性维护措施。数据库系统都具有特定的授权机制，防止非法用户的使用。在多用户操作的情况下，数据库可以进行良好的数据并发处理，维护数据的一致性。

最后，数据库对数据的存取不是以记录为单位的，可以仅操作记录的某些字段，方便了外部应用对数据的操纵。

数据库管理系统

数据库系统是一个多级结构，需要定义各级上的模式，这就需要一组软件提供相应的定义工具；数据库为了保证其中的数据安全和一致性，必须有一套软件来完成相应的控制和管理任务，这样的软件称为数据库管理系统，即 DBMS。

DBMS 的功能随着系统而异，但是通常情况下都包括如下几个方面的功能：

- 数据库描述功能：定义数据库的全局逻辑结构（概念模式）、局部逻辑结构（外模式）以及其它各种数据库对象。
- 数据库管理功能：包括系统控制、数据存储以及更新管理、数据安全性与一致性维护。
- 数据库查询和操作功能：能从数据库中检索信息或者改变信息。
- 数据库建立与维护功能：包括数据写入、数据库重建、数据库结构维护、恢复以及系统性能监视等。

如果以内容来划分 DBMS 的组成，它应该包括下面三个部分：

- 数据描述语言（DDL）以及它的解释程序。
- 数据操纵语言（DML）以及它的解释程序。
- 数据库管理例行程序。

1.1.2　桌面数据库

桌面数据库是一类数据库软件，这些数据库有时又被称为“ISAM 数据库”，因为这些数据库都采用了 ISAM（Indexed Sequential Access Method）文件。目前市场上的桌面数据库包括 Microsoft Access、Microsoft FoxPro 和 Borland Paradox。

桌面数据库将元数据存放在自己的 ISAM 数据文件里，而这些数据文件都是可自描述的，使各种应用程序能够访问桌面数据库里的数据。桌面数据库一般都拥有自己的操作语言和数据类型，包括运行以其操作语言编写的程序的解释程序。可以采用桌面数据库语言来建立数据库应用程序，但是这些经过解释的数据库语言具有极大的局限性，难以建立完整的商业应用。

桌面数据库一般能够提供标准的数据库管理（DBMS）功能，例如数据定义、查询、安全以及维护等，为了提高记录查询速度，桌面数据库要数据增加索引，并使用 ISAM。桌面数据库的 ISAM 文件可以通过局域网（LAN）供网络上的计算机访问，这统称为客户机/服务器（C/S）模式，但是通过 LAN 访问 ISAM 文件的能力和效率是有限的，当 ISAM 数据文件被通过 LAN 访问时，数据要在客户机上进行处理，所有数据和索引都必须从服务器端传送到客户机，造成了应用环境里数据吞吐量的急剧增长。因此说，桌面数据库是专门为个人计算机设计的，是个人计算机数据库应用软件开发的首选数据库。

1.1.3　对象数据库

最原始的数据库技术仅仅在数据文件里存储初始数据，即比特和字节，没有元数据的概念。桌面数据库和对象数据库则同时存放数据和元数据，使数据文件成为可自解释的。对象数据库则进一步在数据文件里存放操作数据的代码。

对象数据库一般都捆绑在特定的编程语言上。C++的对象数据库直接支持 C++语言的

类型系统，可以使用 C++对象数据库在数据库中存储 C++类的实例。下面的代码使用 C++对象数据库存放商品信息。

```
#include <string.h>

// Header file for the object database.
// Management Group object model
#include <odmg.h>

// Derive product class from d_object
// so that the product can persist itself in the database
class Product: public d_object
{
public:
    int iTypeNumber;
    char szPName[96];
    double dPirce;

private:
    d_Ref<Product> next; // 用于遍历数据库中 Product 实例的指针
}

d_Database db; // 全局的对象数据库实例
const char db_name[] = "Product"

void main()
{
    db.open(db_name);   // 打开 Product 对象数据库
    d_Transaction tx;   // 创建一个处理事务并启动
    tx.begin();

    // 在数据库里创建一个新的实例
    Product *product = new (&db, "Product") Product;
    product-> iTypeNumber = 21;
    strcpy(product-> szPName, "Refrigerator");
    product-> dPirce = 2000;
    tx.Commit(); // 将新创建的实例提交到对象数据库
    db.close(); // 关闭对象数据库
}
```

代码开始部分包含一个 odmg.h 的头文件，该文件中包含了 d_Database 类的定义。d_object 使一个 C++基类，可以从该类派生新的类，实现特定数据的操作，Product 类即 d_object 类的派生类。类 Product 具有 d_Ref<Product> next 成员，d_Ref< >是对象数据库供应商提供的一个指针类，通过这个指针类实现对象的引用。

main 函数实现了对象数据库打开、Product 实例的创建与数据赋值、对象提交和对象

数据库关闭等操作，这些代码都不难理解。

C++类使我们能够对复杂实体及其关系进行建模，能够在数据库里存放复杂的 C++类的实例。但是，对象数据库也有其局限性，由于对象数据库同特定的语言紧密集成，所以不便于其它应用程序的访问。另外，在客户机/服务器环境中，对象数据库同桌面数据库一样，具有效率和吞吐量的限制，不能充分利用服务器的处理能力。

1.1.4　关系数据库服务器

关系数据库服务器在某些方面同桌面数据库类似，有自己的编程语言、解释程序和数据类型，也集成数据和元数据。但是关系数据库服务器提供了桌面数据库无法比拟的丰富功能、数据开放性、处理能力以及吞吐能力，同桌面数据库和对象数据库相比，关系数据服务器更适合于客户机/服务器体系结构。

关系数据库服务器可以充分利用高性能服务器的硬件资源，例如大量的 RAM 和高性能的磁盘子系统，将关系数据库服务器安装在 RAID 磁盘系统上，它会充分利用 RAID 驱动程序，提供较大的吞吐能力和可靠性。

关系数据库服务器也有自己的弱点。首先它要比桌面数据库和对象数据库昂贵，与特定商业应用的集成更难；同时可能对硬件有苛刻的要求，还要求数据库管理员定期对系统进行协调和维护。

尽管如此，大型的关系数据库服务器因为其高性能、稳定可靠等优势，成为当前大型商业应用的首选数据库。目前流行的关系数据库服务器有 Oracle、SQL Server、DB2，这些系统各有千秋，需要根据不同的需求进行相应选择。

1.1.5　选择适用的数据库

如何在众多的数据库中选择适合自己应用需求的一个呢？下面的指标值得参考：

- 数据开放性
  其它过程或者应用程序在不访问数据库的源代码时，理解数据文件的能力。
- 复杂数据类型
  处理具有复杂数据实体和关系的应用程序的能力。
- 支持的用户数量
  多个线程、应用程序和用户同时访问数据的能力。
- 性能
  读写数据速度。
- 可伸缩性与能力
  随着数据量的增长，数据库依然保持良好性能的能力。
- 提供基于集合的操作的能力
  是否在其编程模型中提供了基于集合的操作。
- 服务器对基于集合操作的支持
  在服务器端处理数据，而不必将数据传送到客户端进行处理的能力。
- 商业集成的方便性
  是否便于同商业应用程序的集成。

- 数据校验与完整性
  数据库校验数据、确保数据完整性的能力。
- 代码功能比
  所编写的程序代码量与通过执行这些代码所能够获得的数据库能力之比。

1.2　数据库访问技术

1.2.1　概述

数据库是非常复杂的软件，编写程序通过某种数据库专用接口与其通信是非常复杂的工作，为此，产生了数据库的客户访问技术，即数据库访问技术。

数据库访问技术将数据库外部与其通信的过程抽象化，通过提供访问接口，简化了客户端访问数据库的过程。一个好的数据库访问接口就好象程序代码的放大镜，如图 1-1 所示。

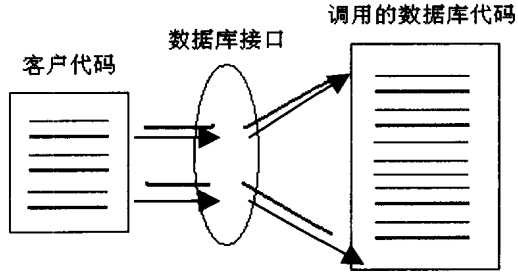

图 1-1　数据库接口的放大镜作用

目前供应商提供的数据库接口分专用和通用两种。专用数据库接口具有很大的局限性，可伸缩性也比较差。通用的数据库接口提供了与不同的、异构的数据库系统通信的统一接口，采用这种数据库接口可以通过编写一段代码实现对多种类型数据库的复杂操作，如图 1-2 所示。

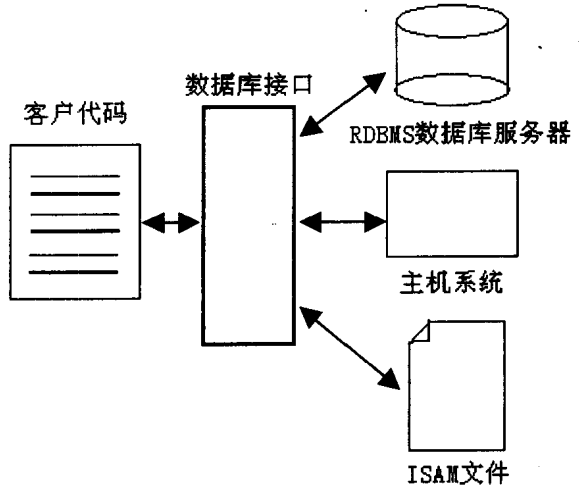

图 1-2　通用数据库接口

目前 Windows 系统上常见的数据库接口包括：

- ODBC（开放数据库互连）
- MFC（Microsoft 基础类）ODBC 类
- DAO（数据访问对象）
- RDO（远程数据对象）
- OLE DB（对象链接嵌入数据库）
- ADO（ActiveX 数据对象）

以下对这些数据库接口作简单介绍。

1.2.2　ODBC API

ODBC 是 80 年代末 90 年代初出现的技术，它为编写关系数据库的客户软件提供了统一的接口。ODBC 只提供单一的 API，可用于处理不同数据库的客户应用程序。使用 ODBC API 的应用程序可以与任何具有 ODBC 驱动程序的关系数据库进行通信。

与其它数据库接口相比，ODBC API 是比较低层的数据库接口，它在相对较低的层次上使客户应用程序可以配置并操作数据库。

由于 ODBC 为关系数据库提供了统一的接口，现在已经被广泛应用，并逐渐成为关系数据库接口的标准。

ODBC 仅限于关系数据库，由于 ODBC 的关系型特性，很难使用 ODBC 与非关系数据源进行通信，例如对象数据库、网络目录服务、电子邮件存储等。

ODBC 提供了 ODBC 驱动程序管理器（ODBC32.DLL）、一个输入库（ODBC32.LIB）和 ODBC API 函数说明的头文件。客户应用程序与输入库连接，以使用 ODBC 驱动程序管理器提供的函数。在运行时，ODBC 驱动程序管理器调用 ODBC 驱动程序中的函数，实现对数据库的操作，ODBC API 的体系结构如图 1-3 所示。

通过 ODBC API 操作数据库的数据库访问方法将在后面详细叙述。

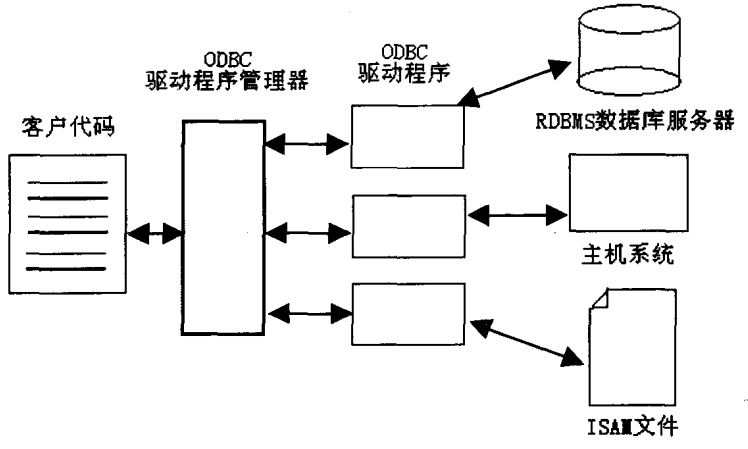

图 1-3　ODBC 体系结构

1.2.3　ODBC 的 MFC 类

ODBC 为关系数据库提供了统一的接口，但是 ODBC API 十分复杂。在 Visual C++中，

MFC 提供了一些类，对 ODBC 进行了封装，以简化 ODBC API 的调用，这些 MFC ODBC 类使 ODBC 编程的复杂型大大降低。

MFC ODBC 类在使用上比 ODBC API 容易，但是损失了 ODBC API 对低层的灵活控制，因此，MFC ODBC 类属于高级数据库接口。

通过 MFC ODBC 类操作数据库的数据库访问方法将在后面详细叙述。

1.2.4　DAO 与 RDO

DAO，即 Data Access Object 的缩写，是一组 Microsoft Access/Jet 数据库引擎的 COM 自动化接口。DAO 直接与 Access/Jet 数据库通信，通过 Jet 数据库引擎，DAO 也可以同其它数据库进行通信，如图 1-4 所示。

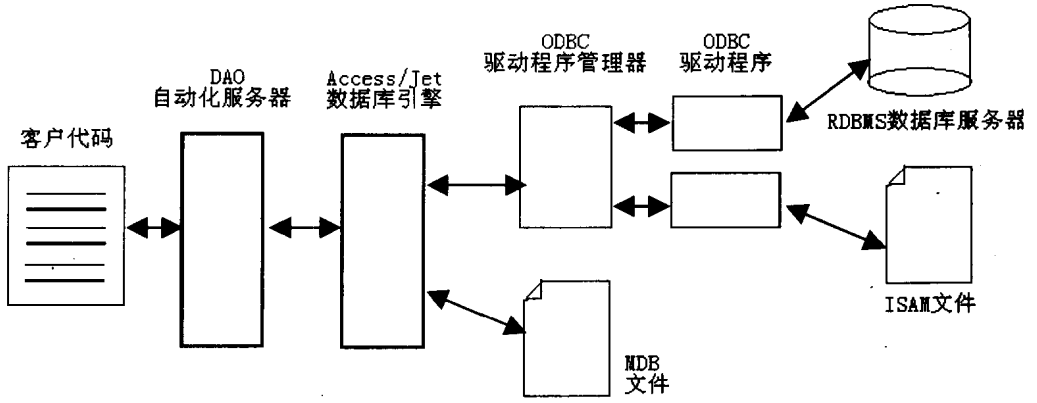

图 1-4　DAO 体系结构

DAO 的基于 COM 的自动化接口提供了比基于函数的 API 更多的功能，DAO 提供了一种数据库编程的对象模型。DAO 的对象模型比一般的 API 更适合于面向对象的程序开发，将一组不关联的 API 函数集成到一个面向对象的应用程序里，一般要求开发人员必须编写自己的一组类来封装这些 API 函数。除了提供一组函数外，DAO 还提供了连接数据库并对数据库进行操作的对象，这些 DAO 对象很容易集成到面向对象应用程序的源代码里。

此外，DAO 还封装了 Access 数据库的结构单元，例如表、查询、索引等，这样，通过 DAO，可以直接修改 Access 数据库的结构，而不必使用 SQL 的数据定义语言（DDL）的语句。

DAO 提供了一种非常有用的数据库编程的对象模型，但是，从图 1-4 可以看出，操作涉及到了许多层的软件。DAO 也提供了访问 Oracle、SQL Server 等大型数据库，当我们利用 DAO 访问这些数据库时，对数据库的所有调用以及输出的数据都必须经过 Access/Jet 数据库引擎，这对于使用数据库服务器的应用程序来说，无疑是个严重的瓶颈。

DAO 同 ODBC 相比更容易使用，但不能提供 ODBC API 所提供的低层控制，因此 DAO 也属于高层的数据库接口。

MFC 对 DAO 的自动化接口做了进一步的封装，叫做 MFC DAO 类。这些 MFC DAO 类都使用前缀 CDao，在后面，我们将详细介绍如何使用 DAO 自动化接口和 MFC DAO 类

对数据库进行操作。

RDO 是 Remote Data Object 的缩写，最初是作为 ODBC API 的抽象，为 Visual Basic 程序员提供的编程对象，因此 RDO 与 Visual Basic 密切相关。由于 RDO 直接使用 ODBC API 对远程数据源进行操作，而不像 DAO 要经过 Jet 引擎，所以，RDO 可以为使用关系数据库服务器的应用程序提供很好的性能。

通过在应用程序里插入 RemoteData 控件，RDO 就可以与 Visual C++一起配合使用。RemoteData 控件是一个 OLE Control，可以被约束到应用程序的界面上，可以通过使用 RemoteData 控件的方法实现对 RDO 函数的调用。

1.2.5　OLE DB 与 ADO

OLE DB 对 ODBC 进行了两个方面的扩展：一是提供了一个数据库编程的 OLE 接口，即 COM；二是提供了一个可用于关系型和非关系型数据源的接口。

OLE DB 提供了 COM 接口。OLE 是 COM 的最初的名字，即使出现了 OLE DB，OLE 仍然被认为是 COM，而 COM 是微软组件技术的基础。

COM 接口同传统的数据库接口相比，例如 ODBC，有更好的健壮性和灵活性，具有很强的错误处理能力，能够同非关系型数据源进行通信。

与 ODBC API 一样，OLE DB 也属于低层的数据库编程接口，OLE DB 结合了 ODBC 对关系型数据库的操作功能，并进行了扩展，可以访问非关系型数据库源。

利用 OLE DB 进行软件开发应该包括两类软件：OLE DB 客户程序（Consumer）和 OLE DB 供应程序（Provider），如图 1-5 所示为 OLE DB 客户程序与供应程序之间的关系。

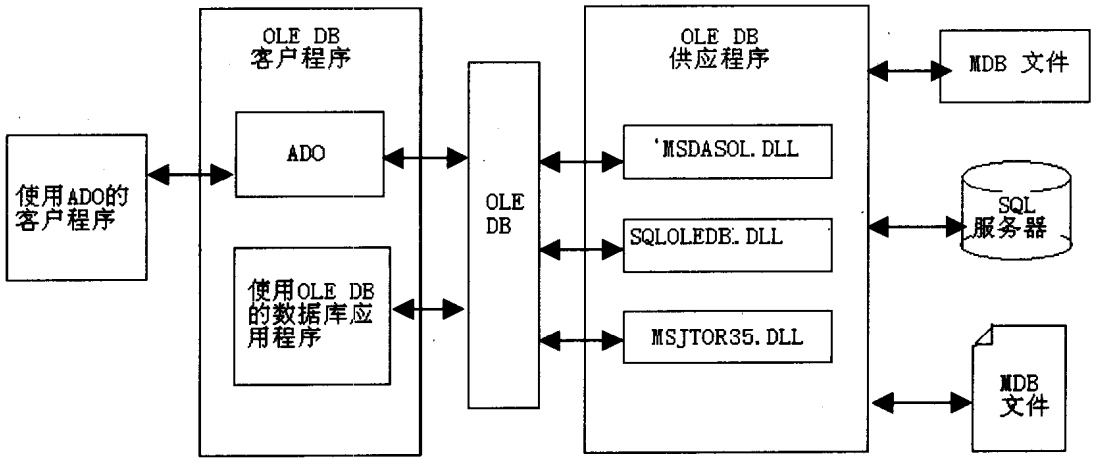

图 1-5　OLE DB 的客户程序和供应程序

OLE DB 客户程序是使用 OLE DB 接口的应用程序，例如，采用 C++编写的使用 OLE DB 连接数据库服务器的所有应用程序都是 OLE DB 客户程序。OLE DB 供应程序是实现 OLE DB 接口并实际与数据库服务器通信的 DLL，在功能上，OLE DB 同 ODBC 驱动程序相同，不过 OLE DB 实现的是 COM 接口，而不是 API 接口。

OLE DB 使我们可以访问任何带有 OLE DB 供应程序的数据源，这些数据源包括电子邮件存储、对象数据库、网络目录及其它非关系数据存储。

OLE DB 是 Windows 操作系统上数据库客户开发的未来发展模式，微软自己的开发也集中在 OLE DB 上，未来所有新的数据库技术也将适用于 OLE DB。OLE DB 的数据库编程技术将在本书第 7 章详细叙述。

ADO 是 ActiveX Data Object 的缩写，它建立在 OLE DB 之上。ADO 实际上是一个 OLE DB 供应程序，使用 ADO 的应用程序要间接地使用 OLE DB。

ADO 提供了一种数据库编程对象模型，类似于 DAO 的对象模型，但比 DAO 有更高的灵活性。ADO 简化了 OLE DB，属于高层的数据库接口。另外同 OLE DB 相比，能够使用 ADO 的编程语言更多。ADO 提供一个自动化接口，使 VBScript 和 JavaScript 等脚本语言可以使用 ADO。本书第 8 章将对 ADO 的数据库编程进行详细介绍。

1.3　数据库操纵语言 SQL

SQL（Structured Query Language）是目前关系数据库领域中主流查询语言，它不仅能够在单机环境下提供对数据库的各种操作访问，而且还能作为一种分布式数据库语言用于客户机/服务器模式数据库应用的开发。

对数据库来说，一个简单的数据获取要求被定义为一个查询，通常需要为此开发特定的查询语言。SQL 的概念是 1974 年由 Boyce 和 Chamberlin 提出的，1986 年成为一个 ANSI 标准，1987 年成为 ISO 标准，目前它广泛应用于数据库管理系统里。

SQL 是一种数据库编程语言，一个 SQL 查询至少包括以下 3 个元素：

- 一个动词，它决定了操作的类型，例如 SELECT。
- 一个宾语，由它指定操作的目标，是一个或者多个字段，是一个或者多个表对象。
- 一个介词短语，由它来决定操作的对象，是数据库的某个对象，例如一个表，或者一个视图。

一个 SQL 语句被传送给一个基于 SQL 的查询引擎，产生查询结果集，结果集以行和列的形式给出。

SQL 语句由命令、从句、运算符和合计函数构成，这些元素组合起来形成语句，用于创建、更新和操作数据库。

1.3.1　SQL 命令

SQL 命令包括以下几种：

- SELECT 命令：用于在数据库中查找满足特定条件的记录，形成特定的查询结果集。这是所有 SQL 语句中最常使用的 SQL 命令。
- CREATE 命令：用于创建数据库的特定对象，如表、索引、视图。
- DROP 命令：用于删除数据库中的特定对象。
- ALTER 命令：用于调整数据库对象的结构。
- INSERT 命令：用于在数据库中向特定表添加一行记录。
- DELETE 命令：用于删除数据库中表的某些记录。
- UPDATE 命令：用于修改数据库中表的某些记录。

1.3.2 SQL 从句

SQL 使用从句来指定查询条件，SQL 从句包括如下几种类型：

- FROM 从句：用于指定从其中选定记录的表的名称。
- WHERE 从句：用于指定所选定记录必须满足的条件。
- GROUP BY 从句：用于指定查询结果集按照特定的列分成不同的组。
- HAVING 从句：用于说明每个组需要满足的条件，一般同 GROUP BY 从句一起使用。
- ORDER BY 从句：用于指定查询结果集按照特定的列排序。

1.3.3 SQL 运算符

SQL 使用的运算符主要有两类：逻辑运算符和比较运算符。

逻辑运算符

逻辑运算符通常出现在 WHERE 从句里，用于组合查询的条件。SQL 语句中经常出现的逻辑运算符主要是如下三种：

- AND：对条件表达式进行“与”操作。
- OR：对条件表达式进行“或”操作。
- NOT：对条件表达式进行“非”操作。

比较运算符

比较运算符用于比较两个表达式的值，从而得到一个条件表达式，下面是常用的 SQL 比较运算符：

- <：小于
- >：大于
- <=：不大于
- >=：不小于
- =：等于
- BETWEEN：指定值的范围
- LIKE：用于模糊查询
- IN：用于指定数据库中的记录

1.3.4 SQL 合计函数

使用合计函数可以对一组数据进行各种不同的统计，它返回用于一组记录的单一值。它能从许多不同行中搜索数据并将它们汇总为一个最终结果。下面是常用的 SQL 合计函数：

- Avg：获得特定列中数据的平均值。
- Count：获得特定的行的计数。
- Sum：获得特定行集合里特定列的数据总和。
- Max：获得特定列的最大值。