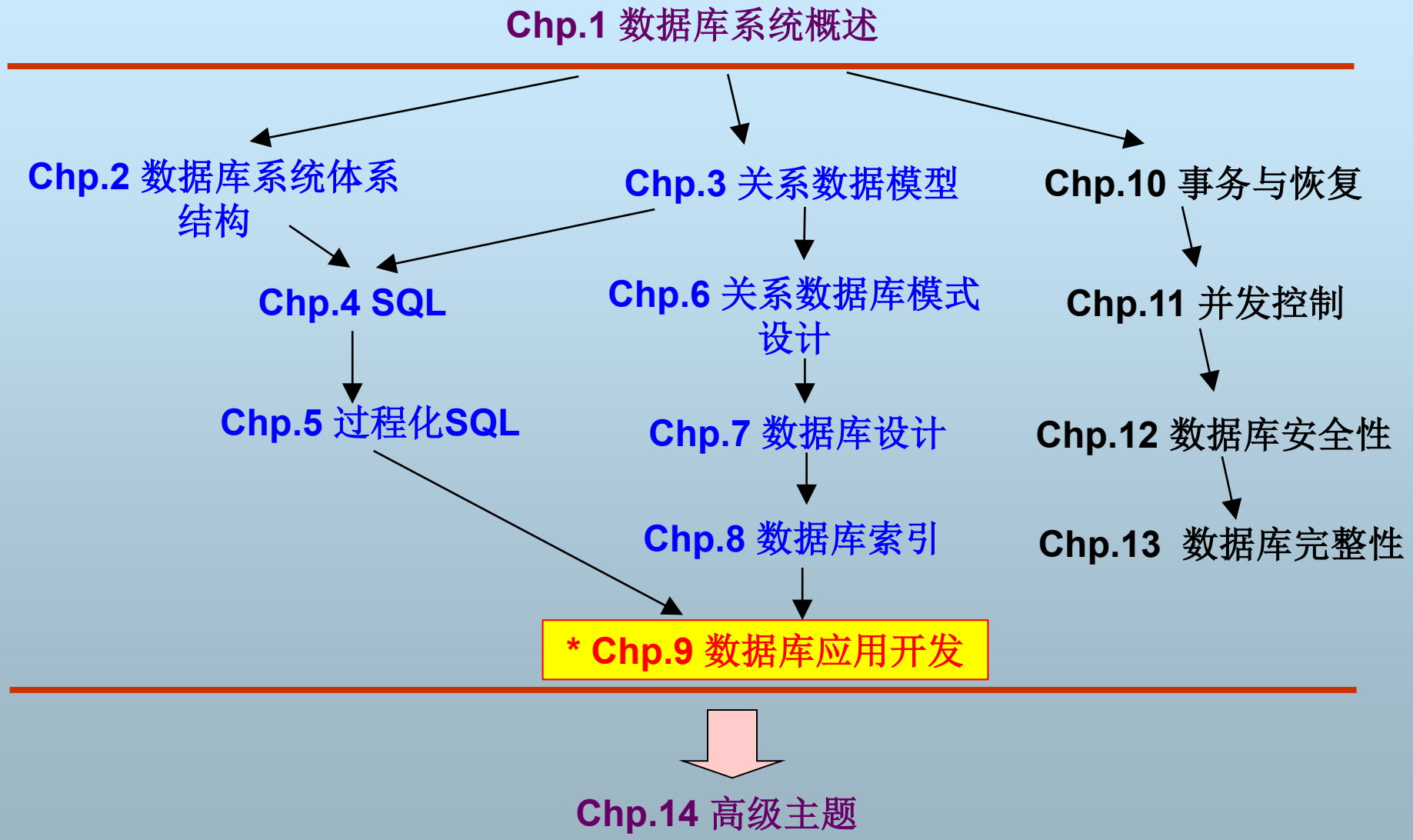
第9章 数据库应用开发

金培权（jpq@ustc.edu.cn）

课程知识结构


Chp.1 数据库系统概述
Chp.2 数据库系统体系结构
Chp.3 关系数据模型
Chp.10 事务与恢复
Chp.4 SQL
Chp.6 关系数据库模式设计
Chp.11 并发控制
Chp.5 过程化SQL
Chp.7 数据库设计
Chp.12 数据库安全性
Chp.8 数据库索引
Chp.13 数据库完整性
* Chp.9 数据库应用开发
Chp.14 高级主题

主要内容

- 数据库应用系统体系结构
- 数据库应用系统开发过程
- 数据库访问编程
- ADO数据库访问示例

一、数据库应用系统体系结构

- 数据库应用的基本需求
- 数据库应用系统体系结构

1、数据库应用的基本需求

- 操作界面服务
  - 数据的输入与显示（如报表显示、图形显示）
- 商业服务
  - 数据处理与检查（如商业规则的检查，如对金额的检查）
- 数据服务
  - 数据储存与维护（如完整性检查）

2、数据库应用系统体系结构

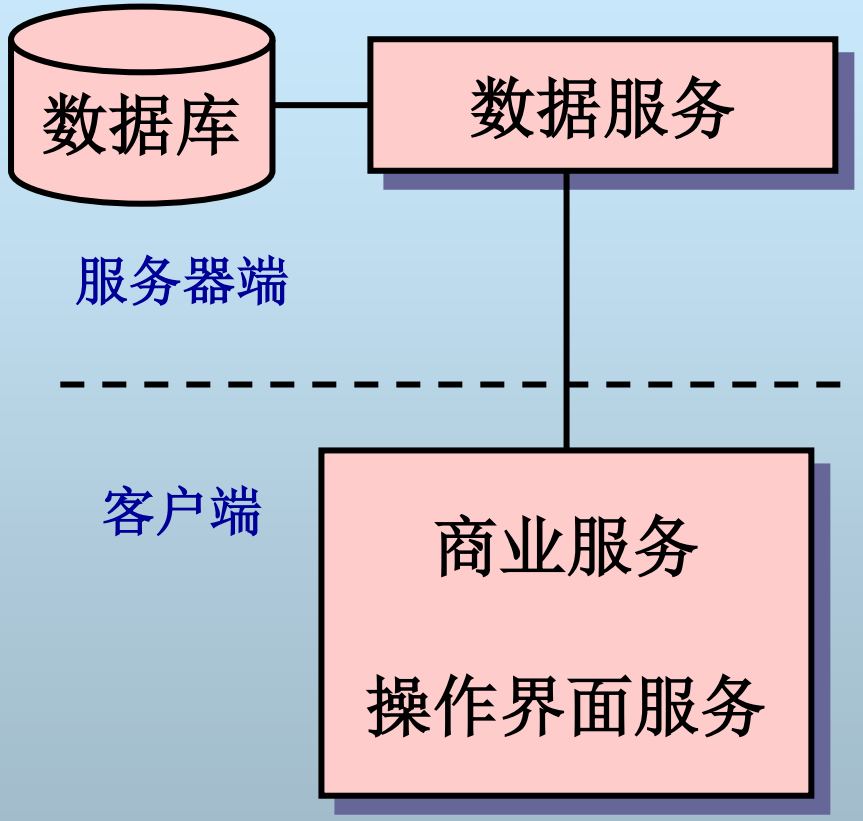
- 根据商业服务层的工作位置不同
  - 以前端为主的两层式结构
  - 以后端为主的两层式结构
  - 三层式处理结构

  } Client/Server (C/S)结构

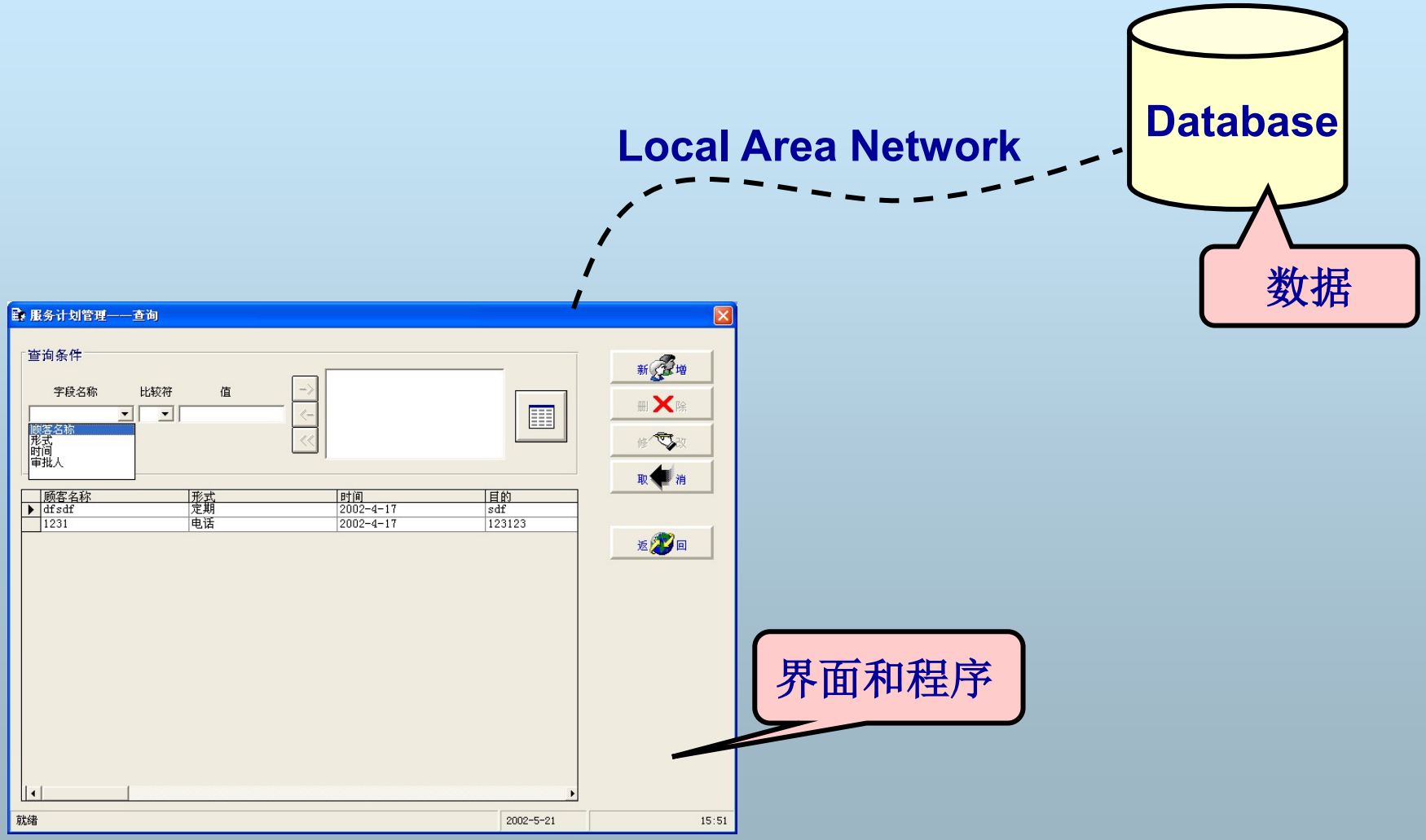
  - 三层Internet处理结构 — Browser/Server (B/S) 结构
  - 多层Internet处理结构 — N－Tier 结构
  - 混合结构

（1）以前端为主的两层式结构

- 传统的开发方法
- 后端服务器只提供数据服务
- 商业服务由前端工作站完成
- 开发和调试容易
- 当用户数增加时，网络数据传送负担加重

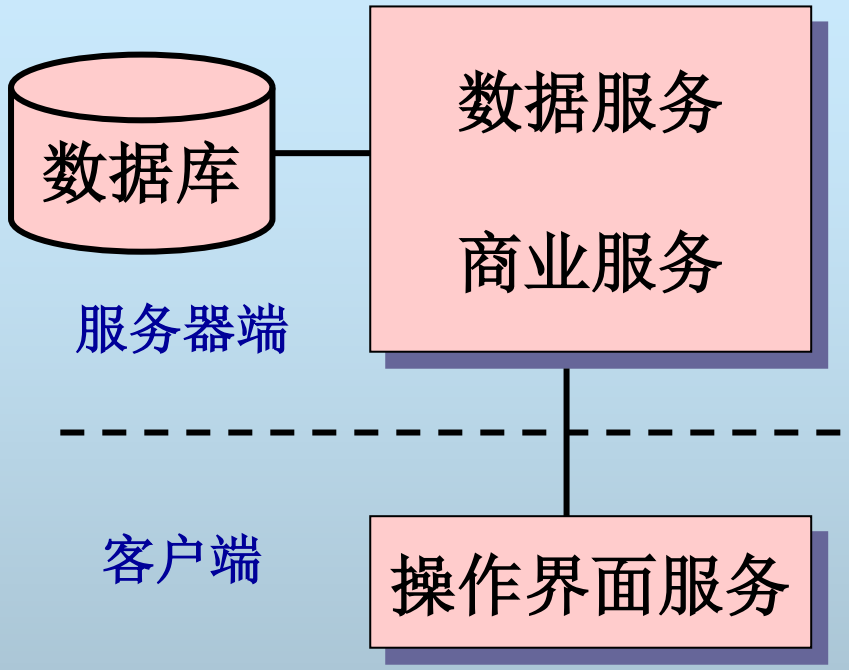


（1）以前端为主的两层式结构

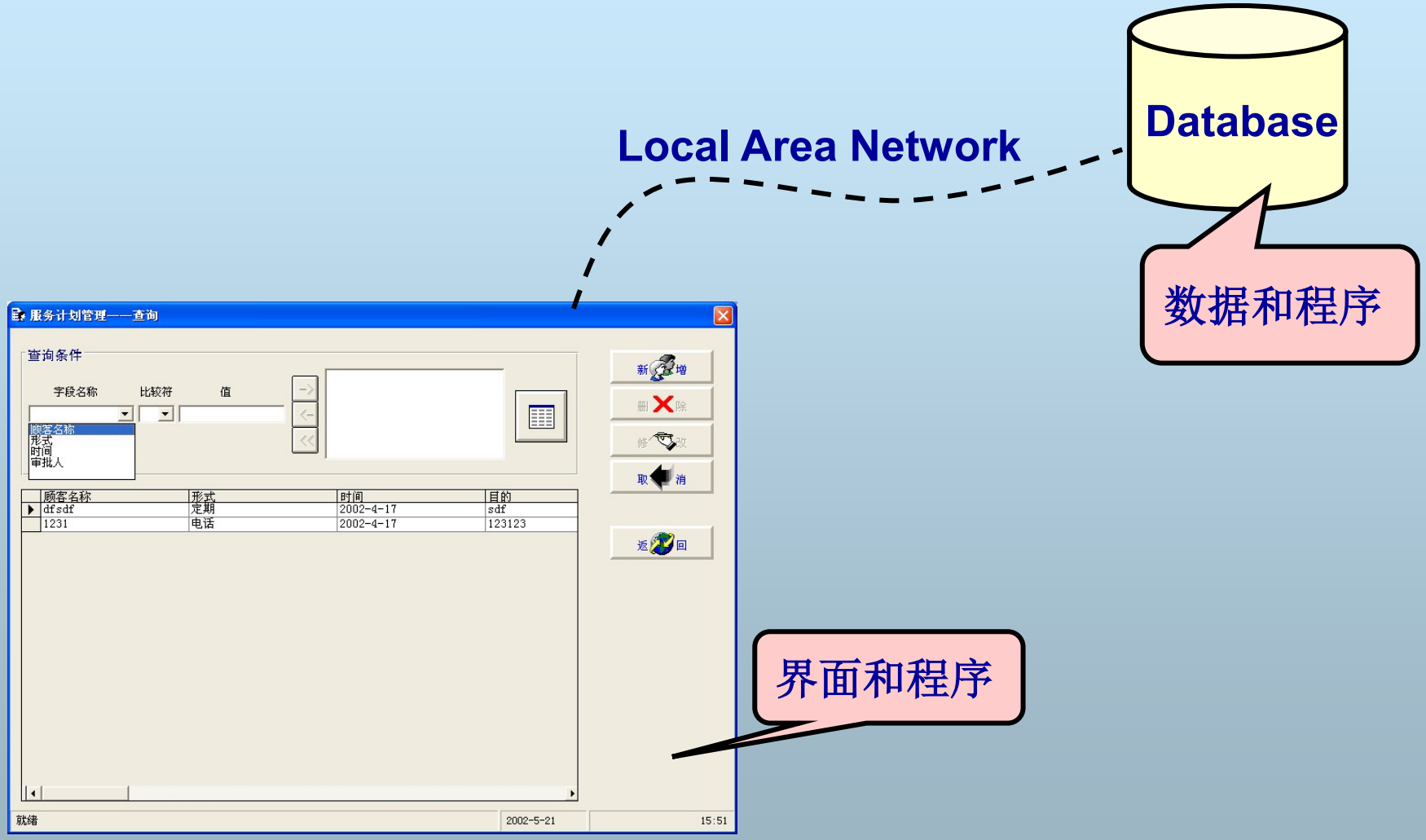


（2）以后端为主的两层式结构

- 后端服务器提供数据服务和商业服务
- 借助存储过程和触发器来完成商业服务
- 开发和调试受限制
- 减少了网络数据传送

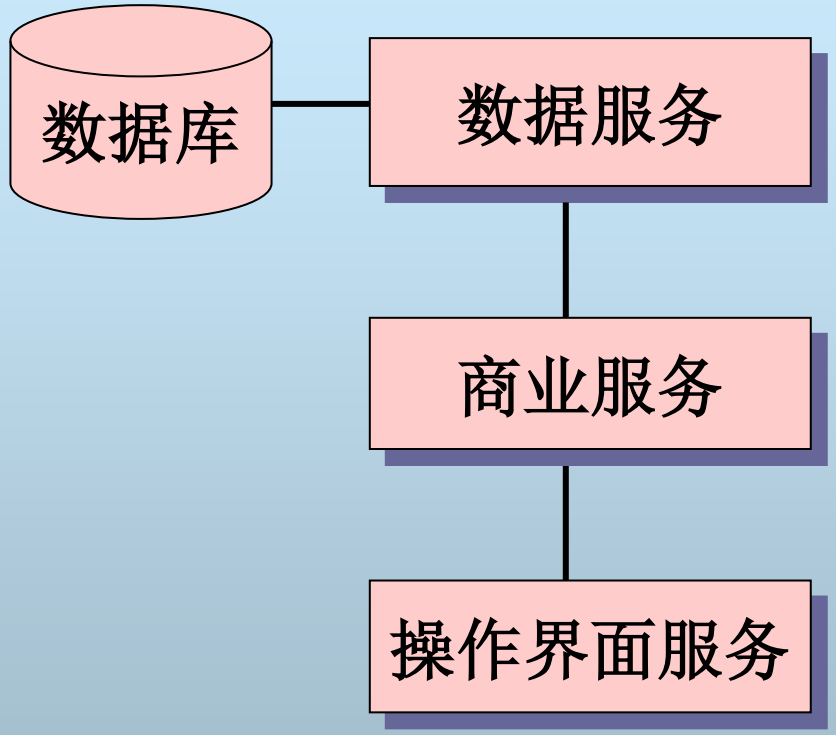


（2）以后端为主的两层式结构

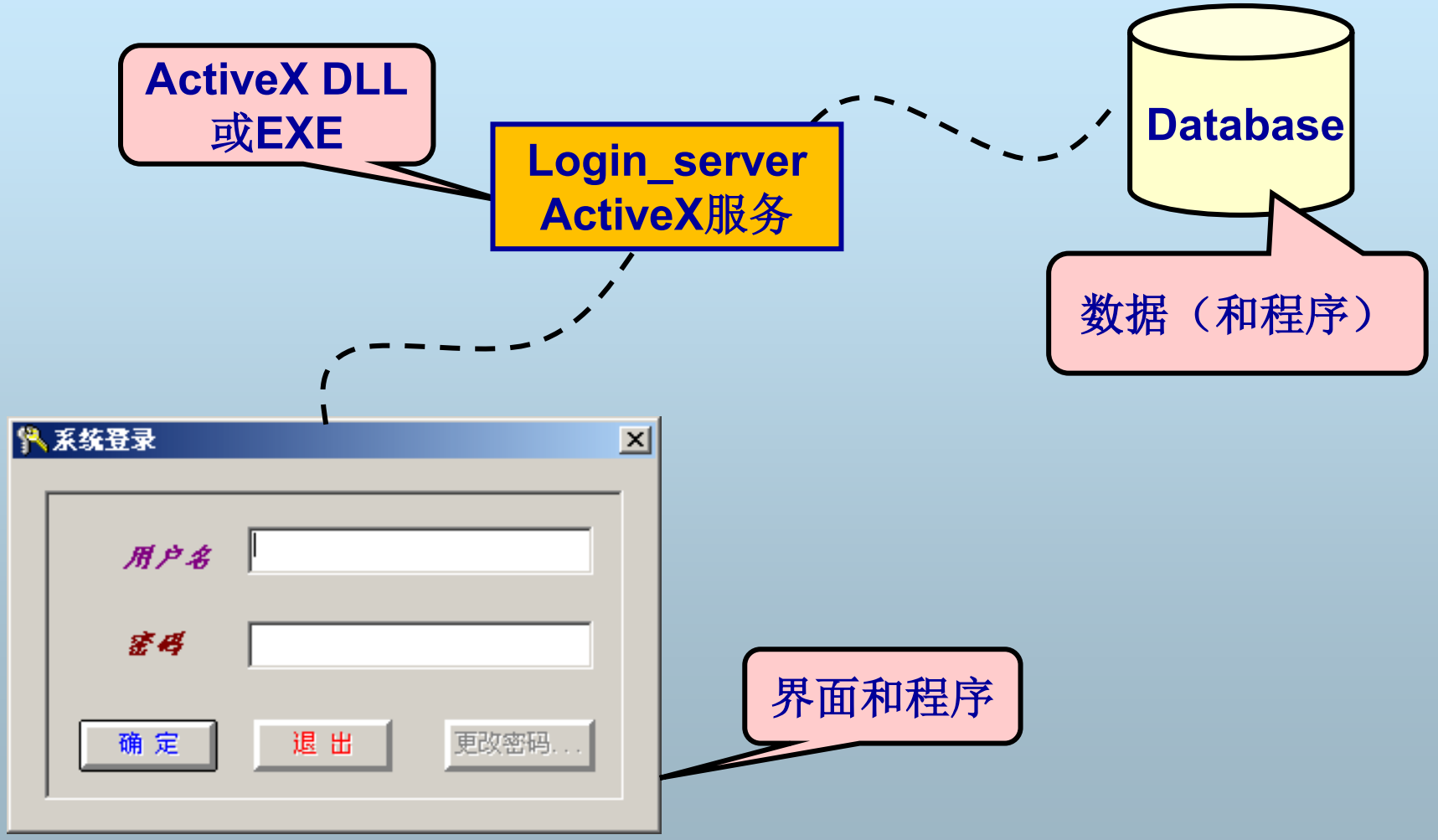


（3）三层式处理结构

- 商业服务独立运行（如ActiveX 服务器）
- 可以位于不同服务器，也可以和数据库服务器同一主机
- 可以分别减轻前后端的工作负荷
- 开发和调试相对复杂

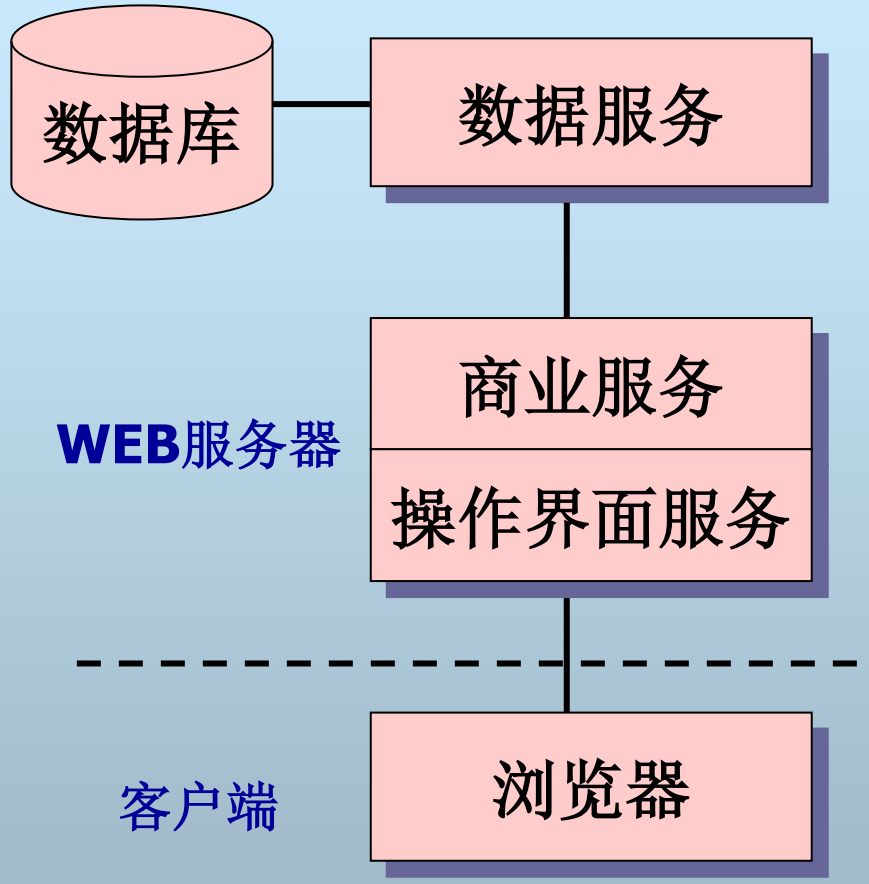


（3）三层式处理结构

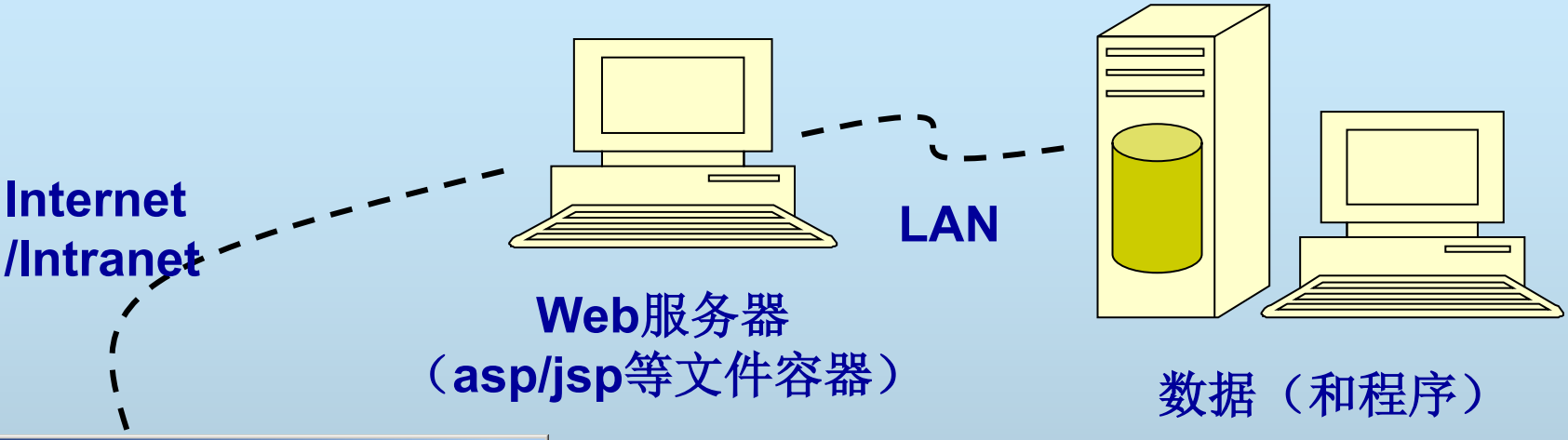

ActiveX DLL
或EXE
Login_server
ActiveX服务
Database
数据（和程序）
系统登录
用户名
密码
确 定
退 出
更改密码...
界面和程序

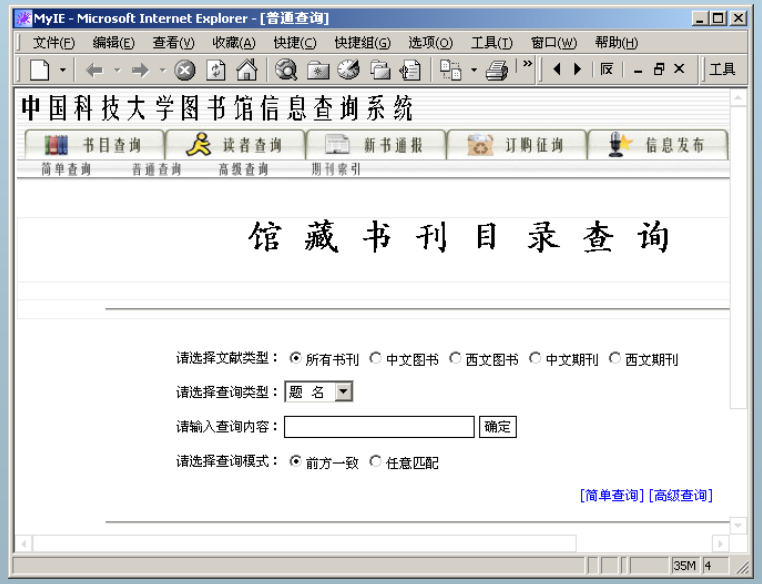
（4）三层Internet处理结构

- 三层式设计结构
- 将操作界面服务分割到浏览器和WEB服务器上
- 商业服务仍然可有多种安排方式
- 系统可以跨平台运行
- 客户端管理容易



（4）三层Internet处理结构

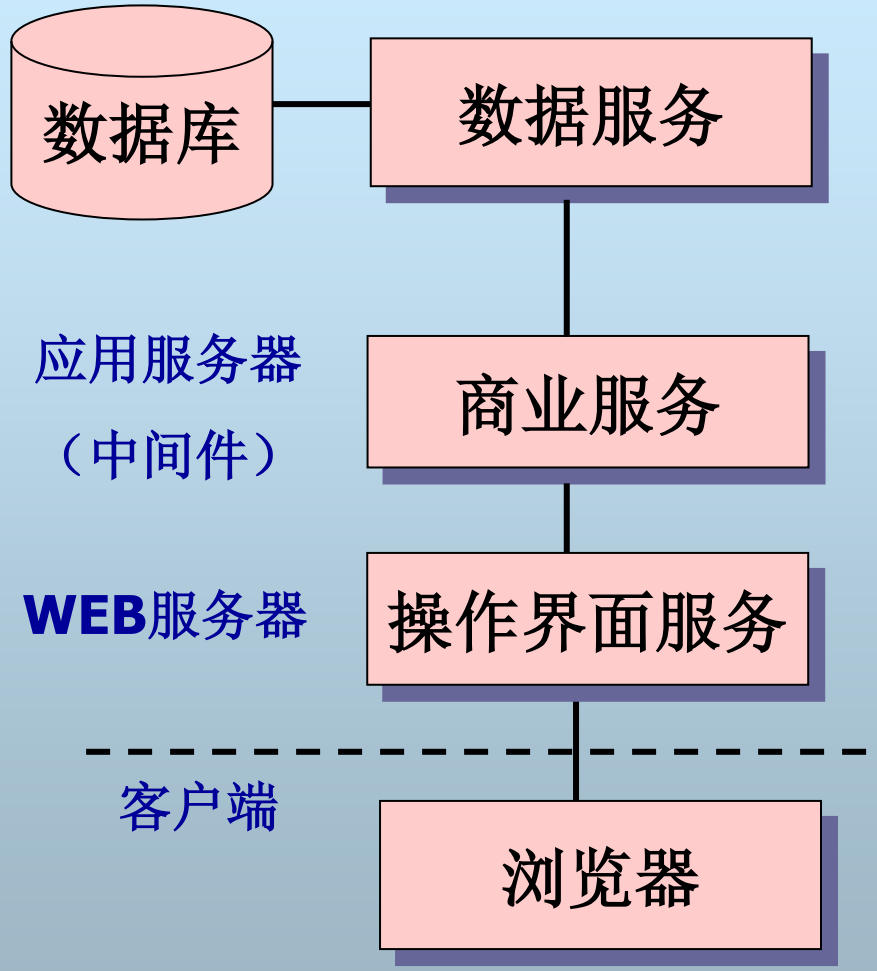




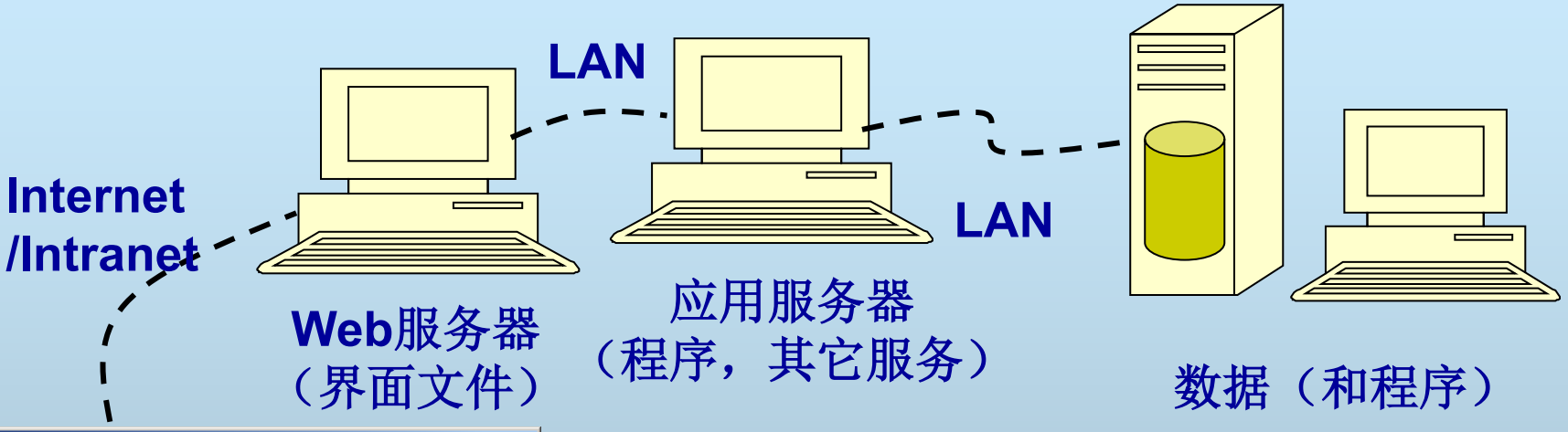
html

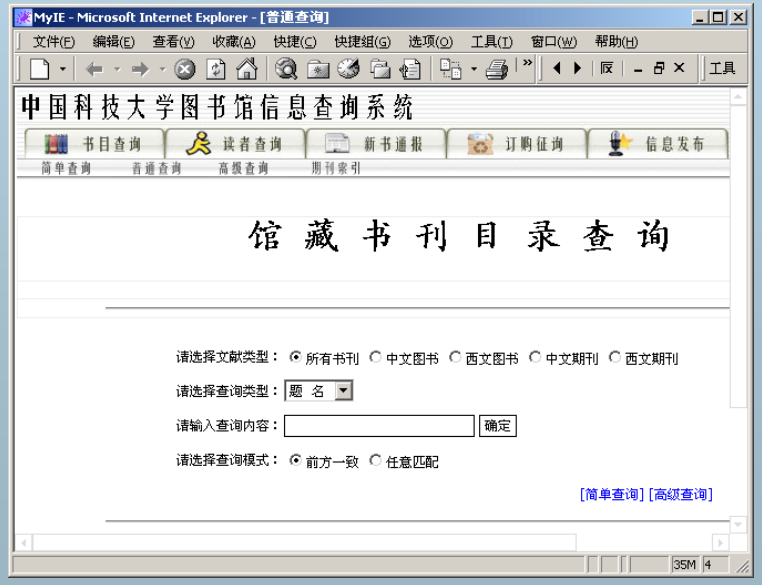
（5）多层Internet处理结构

- 多层式设计结构
- 将商业服务放到应用服务器，实施负载均衡等功能
- WEB服务器负责操作界面服务
- 系统独立性高
- 可以跨平台运行
- 客户端管理容易
- 服务器端部署和管理较复杂



（5）多层Internet处理结构

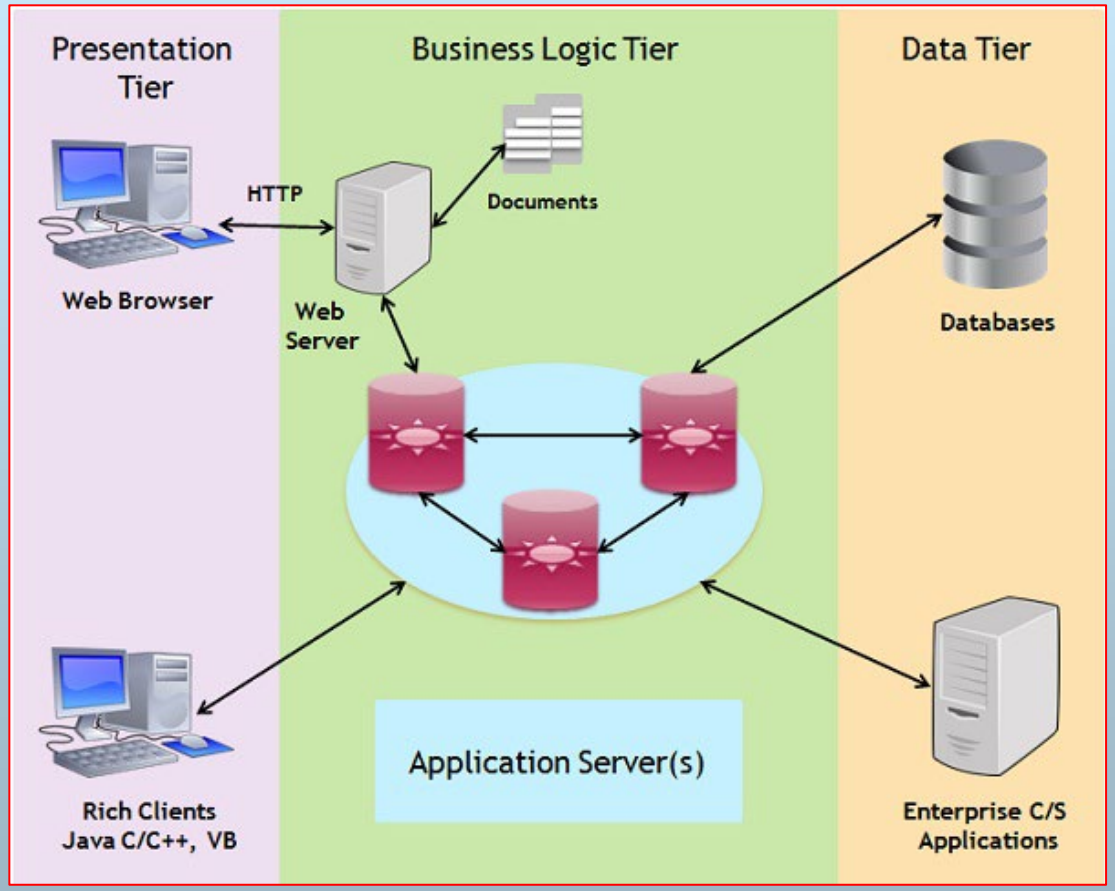




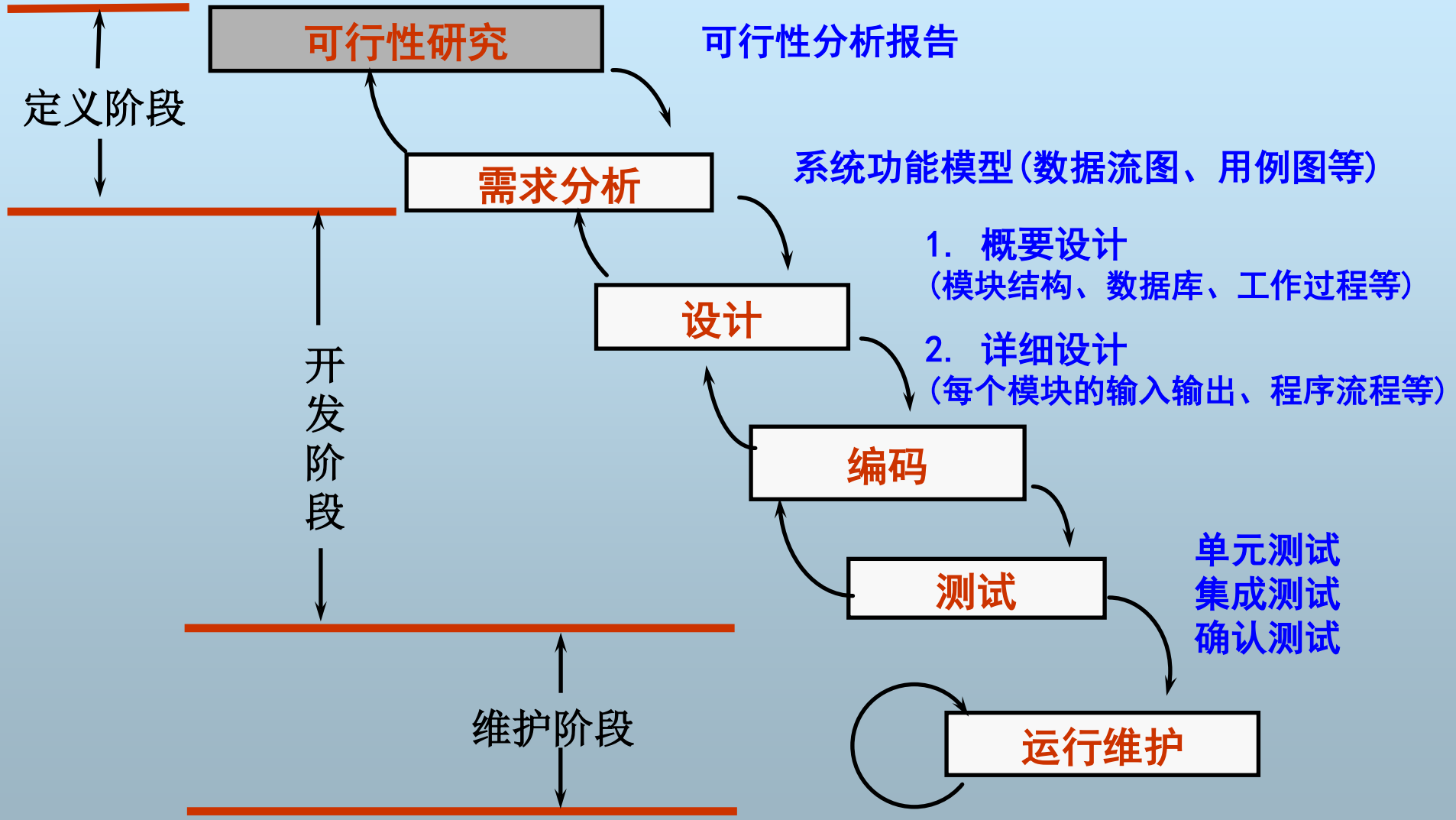
html

（6）混合结构

- 实际系统开发时可根据需求采取混合结构
- 内部管理功能
  - C/S结构
  - 保证访问的可控性和安全性
- 外部功能
  - B/S结构
  - N-Tier结构
    - 如果访问负载较大
    - 还需要邮件服务、FTP服务等其它功能



二、数据库应用系统分析与设计

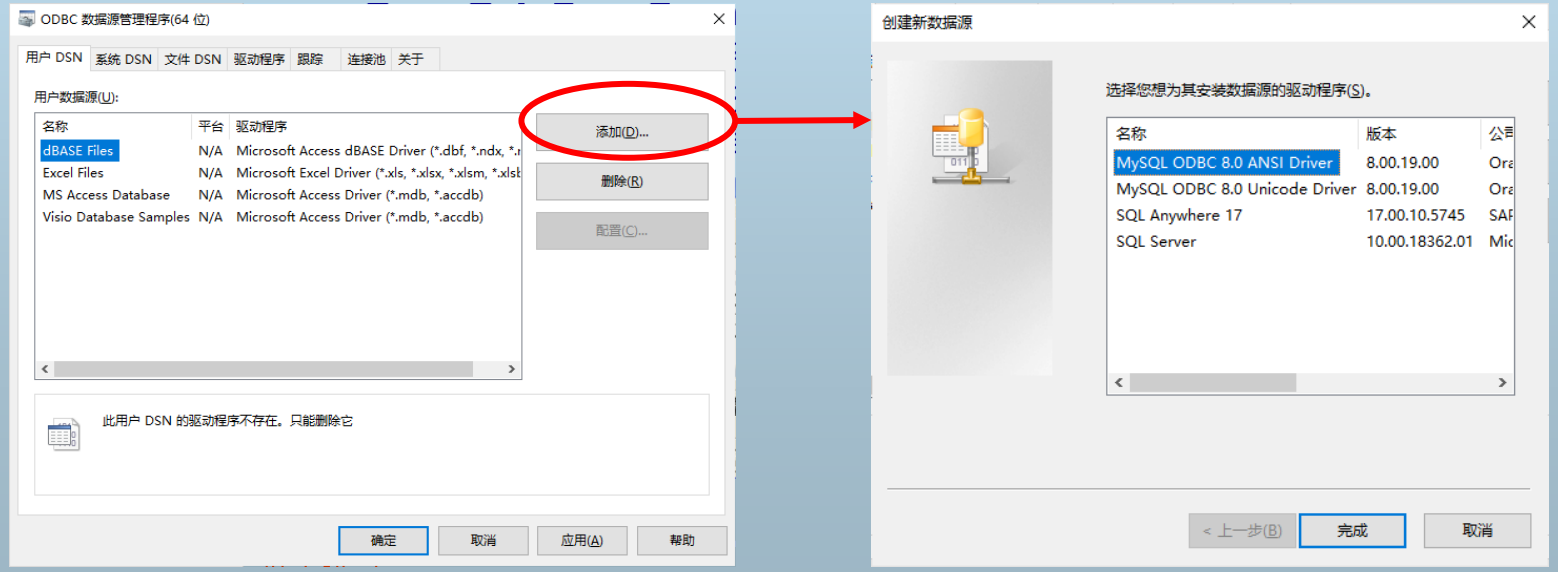


三、数据访问编程

- 数据库访问方法
- 典型的数据库应用结构
- 数据库基本操作
- 数据库应用编程过程

1、数据库访问方法

- 早期的数据库访问方法ODBC
  - Open DataBase Connectivity（ODBC）是微软公司倡导的、当前被业界广泛接受的、用于数据库访问的应用程序编程接口（API），它底层使用结构化查询语言（SQL）作为其数据库访问语言。



1、数据库访问方法

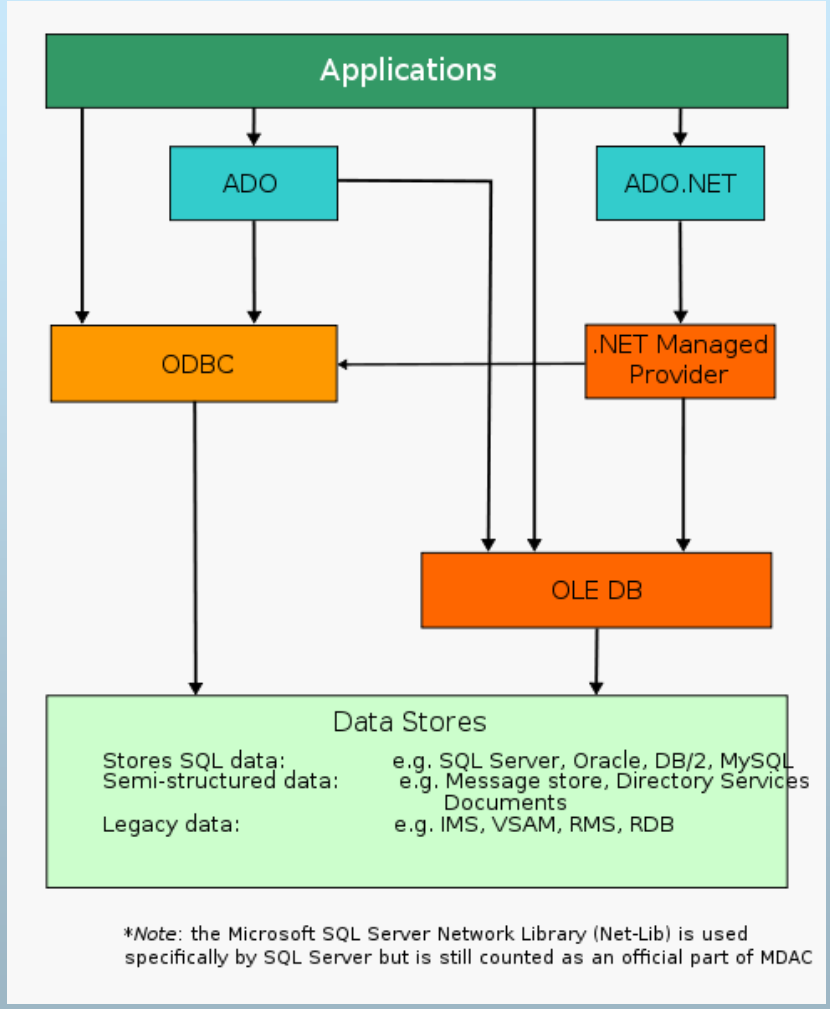
- Java访问数据库的专用接口JDBC
  - JDBC（Java DataBase Connectivity）是Java与数据库的接口规范，JDBC定义了一个支持标准SQL功能的通用低层的应用程序编程接口（API）。底层通过SQL访问数据库。
  - JDBC的设计思想与ODBC类似，但JDBC是与Java语言绑定的，所以不能用于其它编程语言。

1、数据库访问方法

- 目前流行的数据库访问模型ADO
  - ActiveX Data Objects，即ActiveX数据对象。
  - ADO是微软新的通用数据存取框架。它包含了ODBC、数据库访问对象（DAO）、远程数据对象（RDO）及几乎所有其他数据存取方式的全部功能。
  - 用户可以利用ADO连接SQL Server、Oracle及其他的数据源。

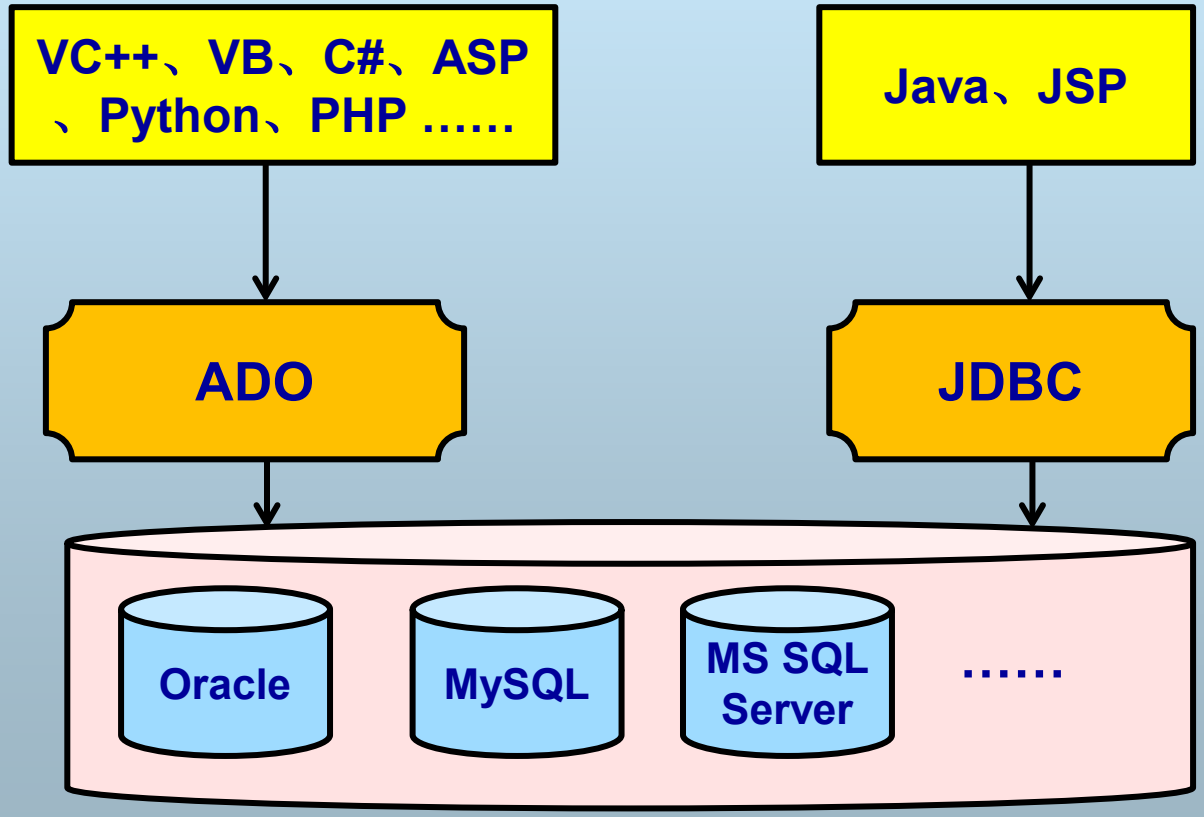
1、数据库访问方法

- ADO通过OLE DB驱动或ODBC访问数据源，不仅支持SQL数据库访问，也支持Excel、Text等非结构化数据访问
- 由于ADO能够以统一的方式连接各种数据源，因此成为一种与编程语言独立的数据库访问模型
- ADO.Net是工作在.Net Framework上的数据库访问模型，功能与ADO类似



1、数据库访问方法

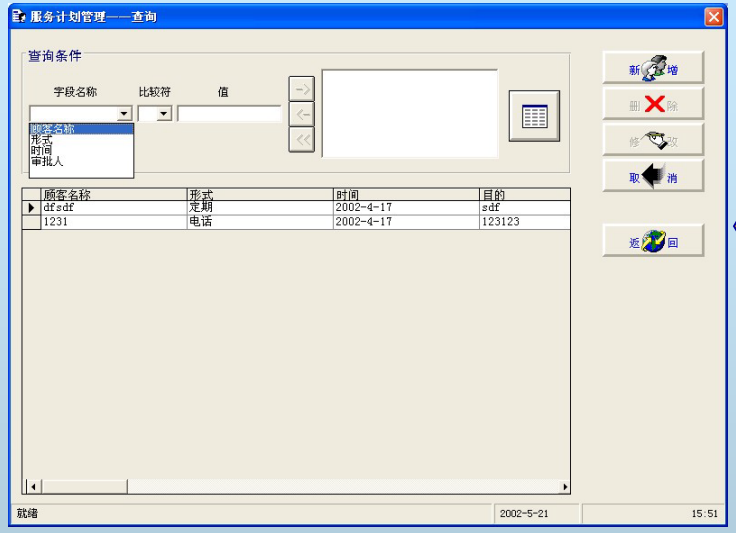
总体的数据库访问模型

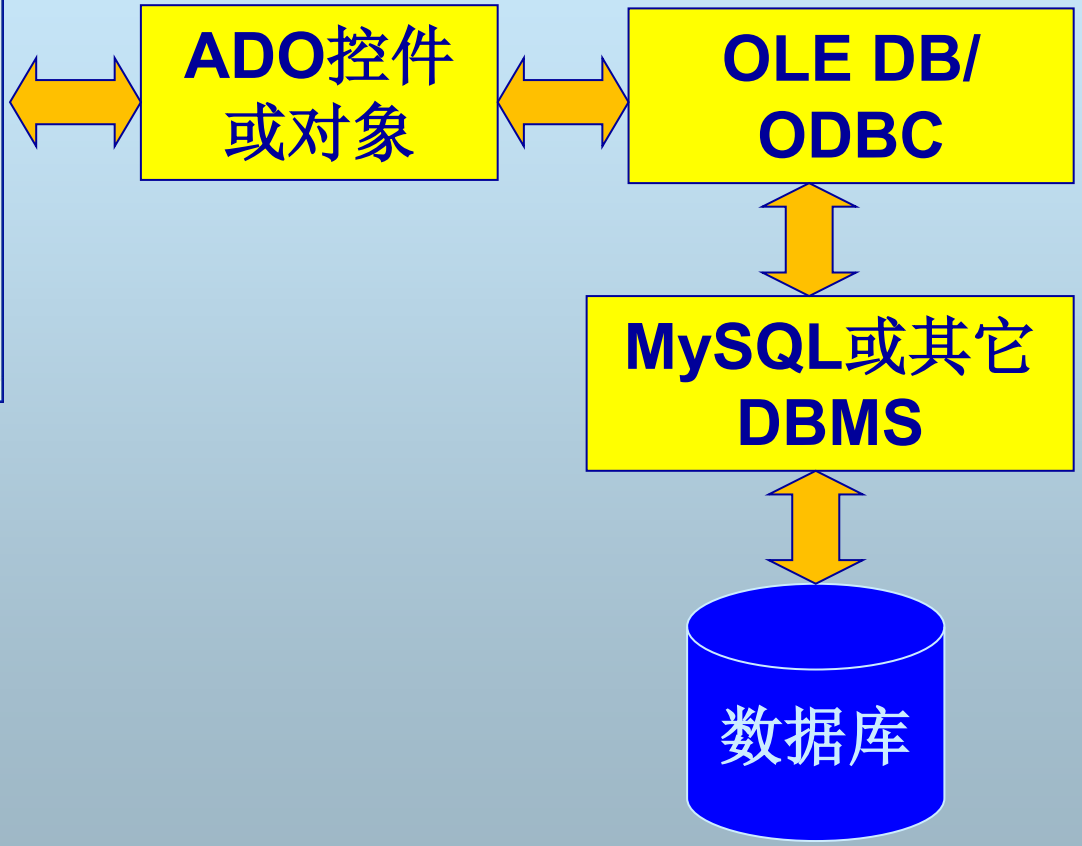


 # 编程语言排行榜2024 年 5 月

排名	编程语言	流行度	对比上月
1	Python	16.33%	-0.08%
2	C	9.98%	-0.23%
3	C++	9.53%	-0.23%
4	Java	8.69%	-0.25%
5	C#	6.49%	-0.28%
6	JavaScript	3.01%	0.12%
7	Visual Basic	2.01%	0.31%
8	Go	1.60%	-0.25%
9	SQL	1.44%	-0.17%
10	Fortran	1.24%	-0.23%

2、典型的数据库应用结构

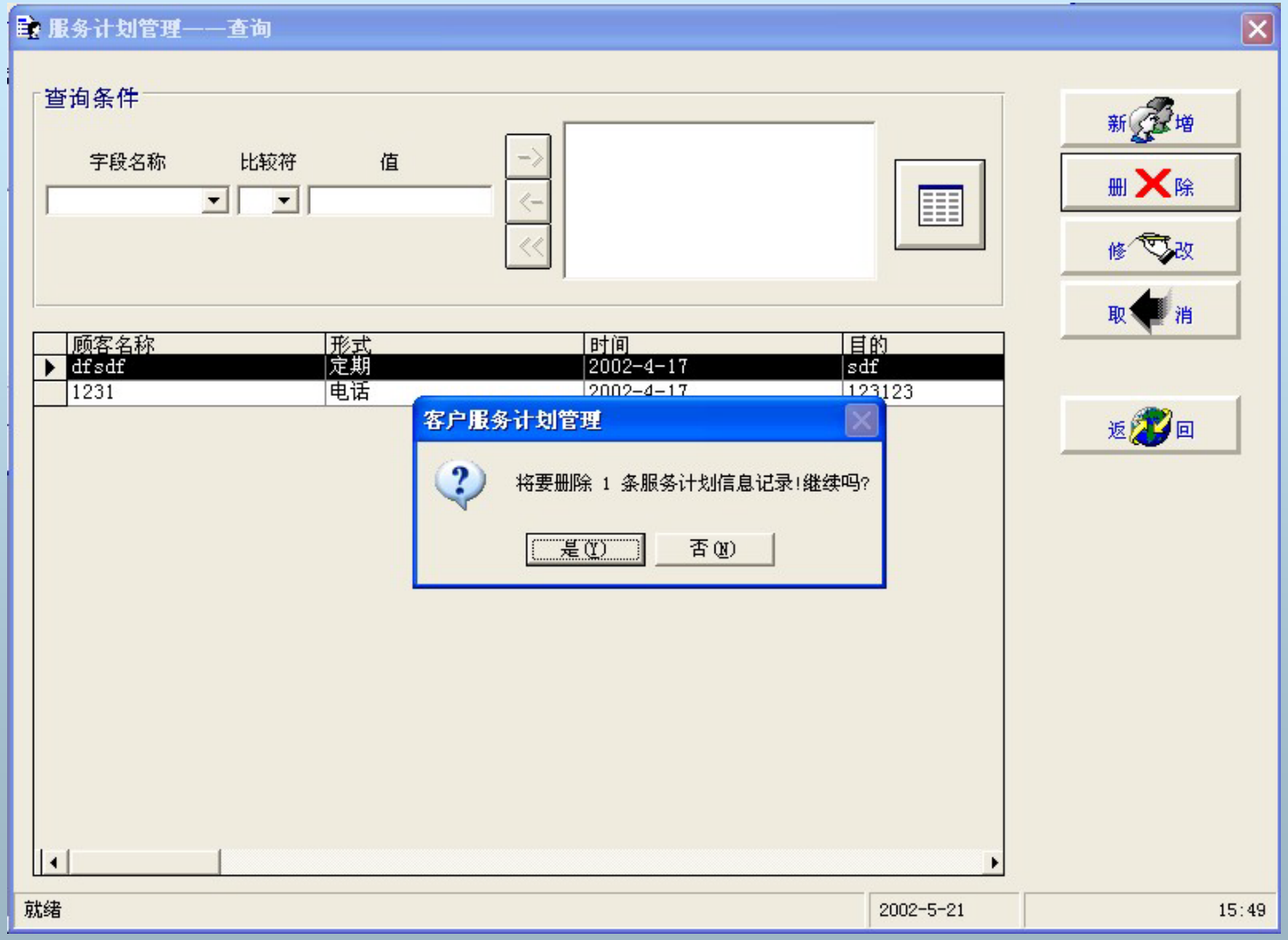




3、数据库基本操作

- 界面和功能围绕数据库中数据操作进行设计
  - 数据的增加功能
  - 数据的修改功能
  - 数据的删除功能
  - 数据的查询功能

3、数据库基本操作


服务计划管理——查询
查询条件
字段名称
比较符
值
新增
删除
修改
取消
返回
顾客名称
形式
时间
目的
dfsdf
定期
2002-4-17
sdf
1231
电话
123123
客户服务计划管理
将要删除 1 条服务计划信息记录!继续吗?
是(Y)
否(N)
就绪
2002-5-21
15:49

4、数据库应用编程过程

- 数据库应用的功能往往以数据的管理为核心，因此编程应以实现数据管理功能为主。基本的过程包括
  - 数据管理界面设计
    - 增、删、改、查界面设计，由于查询是数据库应用中最常用的功能，因此界面往往以查询界面为主展开设计
  - 数据管理功能的编程实现
    - SQL是应用与数据库的唯一接口
    - 一般通过高层的编程接口如ADO实现数据库操作

4、数据库应用编程过程

- 简单示例

4、数据库应用编程过程

- 数据访问的一般步骤
  - 建立数据库连接
  - 声明数据库编程接口对象
  - 通过对象实现数据访问功能
  - 释放对象
  - 关闭连接

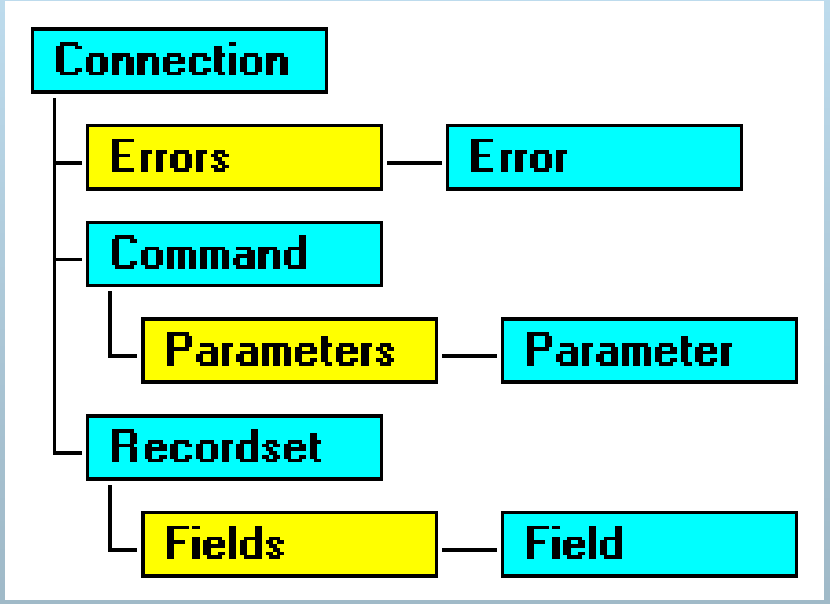
四、ADO数据库访问示例

- ADO数据访问模型
- 增加记录（Create）
- 查询记录（Read）
- 删除记录（Update）
- 修改记录（Delete）

“CRUD”

1、ADO数据访问模型

- ADO通过对象和集合来实现数据库操作
- 黄色框表示集合



Connection对象

- Connection 对象代表了打开的、与数据源的连接。
- 定义（以VB为例）
  - Dim cnn as New ADODB.Connection
- 主要属性
  - ConnectString
  - CursorLocation

Connection对象示例

- 主要的方法
  - Open，Close
  - Execute `可执行SQL语句
  - BeginTrans，CommitTrans，RollbackTrans `用于事务编程
- 示例

```
Dim cnn as New ADODB.Connection
Cnn.Connectstring= "Provider=OraOLEDB.Oracle; Data Source=ORCL; User ID=users; Password=abcd;"
` Cnn.Connectstring="DSN=Mysql; SERVER=192.168.1.11;UID=root;PWD=root;PORT=3306;DATABASE=mydb"
Cnn.CursorLocation=adUseClient
Cnn.Open
```

通过 OLEDB

通过 ODBC

Command对象

- Command 对象定义了将对数据源执行的指定命令。
- 可使用 Command 对象查询数据库并返回 Recordset 对象中的记录
- 执行某个存储过程

Command执行SQL语句

- 主要的属性
  - **ActiveConnection：所使用的Connection**
  - **CommandText：定义命令（例如，SQL 语句）的可执行文本。**
- 主要的方法
  - **Execute**
  - **CreateParameter**
- 示例


```
Dim cmm as New ADODB.Command
Dim rst as New ADODB.Recordset
Cmm.ActiveConnection=cnn
cmm.CommandText="select * from s where b='asas' "
Set rst=cmm.Execute()
```

Command执行存储过程

- 常用方式
- 假设已经写好了一个存储过程samp
  - 计算给定部门的员工的人数和平均工资
  - 输入参数：部门名称
  - 输出参数：人数和平均工资

调用存储过程samp，计算给定部门的员工的人数和平均工资

```
Dim cmm as New ADODB.Command
Set cmm.ActiveConnection= cnn ‘cnn为数据库连接，在此假设其已建立
Cmm.CommandText=“samp” ‘存储过程名
Cmm.CommandType=adCmdStoredProc ‘设为存储过程
‘为存储过程调用添加参数
cmm.Parameters.Append cmm.CreateParameter("Return", adInteger, adParamOutput, 4, 0)
cmm.Parameters.Append cmm.CreateParameter(“DeptName", adVarChar, adParamInput, 50, "")
cmm.Parameters.Append cmm.CreateParameter(“EmpCount", adInteger, adParamOutput, 4, 0)
cmm.Parameters.Append cmm.CreateParameter(“AvgSalary", adNumeric, adParamOutput, 8, 0)
‘传递参数
Cmm.Parameters(“DeptName”)= txtDept.Text ’假设输入的部门名在txtDept中
Cmm.Execute ’执行
If cmm.Parameters(“Return”)= -20001 then
    Msgbox “部门不存在”
    Exit Sub
Elseif cmm.Parameters(“Return”)=-20002
    Msgbox “部门没有员工”
    Exit sub
End if
Msgbox “员工数＝ “ & cmd.parameters(“EmpCount”)
Msgbox “员工平均工资＝ “ & cmd.parameters(“AvgSalary”)
```

Connection和Command总结

- Connection一般用于建立数据库连接
  - 数据库应用的最终操作对象一般是记录集（Recordset）
- Command一般可用于执行某个存储过程
  - 对于“Select”、”Insert”等SQL语句，一般使用Recordset对象来实现。
  - 某些特殊情况下，比如要批量导入数据时可以考虑用Command执行SQL

Recordset 对象

- Recordset 对象表示的是来自基本表或命令执行结果的记录全集。
- 数据库应用中最常使用的ADO对象
- 可以完成针对记录的所有操作

Recordset对象的属性

- BOF和EOF
- Source: 表示所基于的基本表或SQL语句
- CursorLocation
- CursorType
  - 游标类型。一般使用adOpenKeyset（仅修改可见）或adOpenDynamic（全部可见）
- LockType
  - 指示编辑过程中对记录使用的锁定类型，一般adLockOptimistic，表示仅在Update时锁定
- Recordcount
  - 记录总数

Recordset的方法

- Open
- Close
- Addnew
- Update
- Delete
- Movefirst、MoveNext……
- Requery

操作数据库的一般过程

- 创建Connection(Open)
- 打开Recordset(Open)
- 使用Recordset的Addnew、Update、Delete、Move等方法对数据进行增、删、改
- 查询，修改Source然后再Open即可。

Open

- *recordset*.**Open** *Source, ActiveConnection, CursorLocation, CursorType, LockType, Options*
- 基于已有的Connection的Open
  - rst.Open "Employees", cnn, adUseClient, adOpenKeyset, adLockOptimistic, adCmdTable
- 不使用已有的Connection直接打开Recordset
- 将cnn换成Connectstring的内容即可。

记录的添加：AddNew

```
Dim cnn as New ADODB.Connection
Cnn.Connectstring=…
Cnn.Cursorlocation=adUseClient
Cnn.Open
Dim rst as New ADODB.Recordset
rst.Open “Employees”, cnn, adUseClient, adOpenKeyset, adLockOptimistic, adCmdTable
rst.Addnew
rst.Fields(“Name”)=txtName.Text
……
rst.Update
rst.Close
```

AddNew的过程分析

- 打开数据库连接(Connection对象可以是局部对象，也可以是全局对象)
- 打开Recordset，一般是一个基本表
- rst.Addnew
- 将值赋给字段
- rst.Update
- rst.Close

记录的删除

- **根据输入的EmployeeID值(txtID)删除相应记录**

```
Dim rst as New ADODB.Recordset
rst.Open "select * from Employees where EmployeeID='" & txtID.text & "'", cnn, adUseClient, adOpenKeyset, adLockOptimistic, adCmdText
If Not(rst.EOF and rst.BOF) then
    rst.delete
Else
    Msgbox "记录不存在"
End if
rst.Close
```

删除过程分析

- 使用rst.Open方法根据给出的删除条件创建一个记录集，该记录集包含了要删除的所有记录
- 检查记录集是否为空（因为一般一次只删除一条记录，因此要按主键去Open记录集）
- rst.Delete
- rst.Close

记录的修改

- **设将EmployeeID值为‘100’的记录的Name修改为‘aaaa’，Salary修改为2000**

```
Dim rst as New ADODB.Recordset
rst.Open "select * from Employees where
    EmployeeID='100'", cnn, adUseClient, adOpenKeyset,
    adLockOptimistic, adCmdText
rst.Fileds("Name")="aaa"
rst.Fields("Salary")=2000
rst.Update
rst.Close
```

修改过程分析

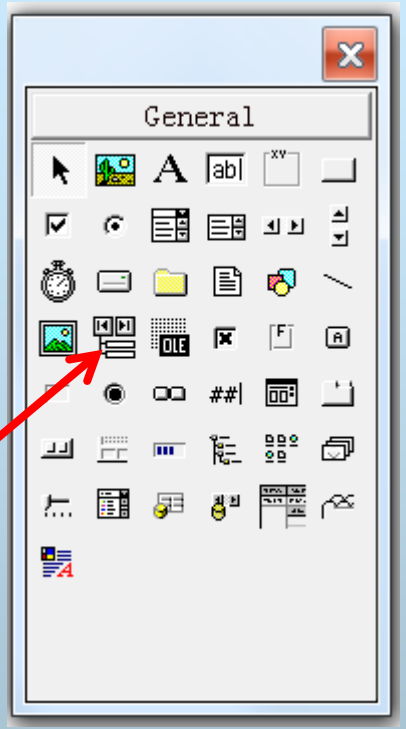
- 使用Open定位要修改的记录（即创建最多只有一条记录的记录集）
- 将新值赋給字段
- 调用Update方法
- Close记录集

记录的查询

- 如果在程序里查询（不需要显示），可以直接使用Recordset对象根据生成的SQL语句Open即可
- 如果需要在界面上显示（一般是表格），则可以使用ADO Data控件和数据绑定控件
  - ADO Data控件
  - 数据绑定控件

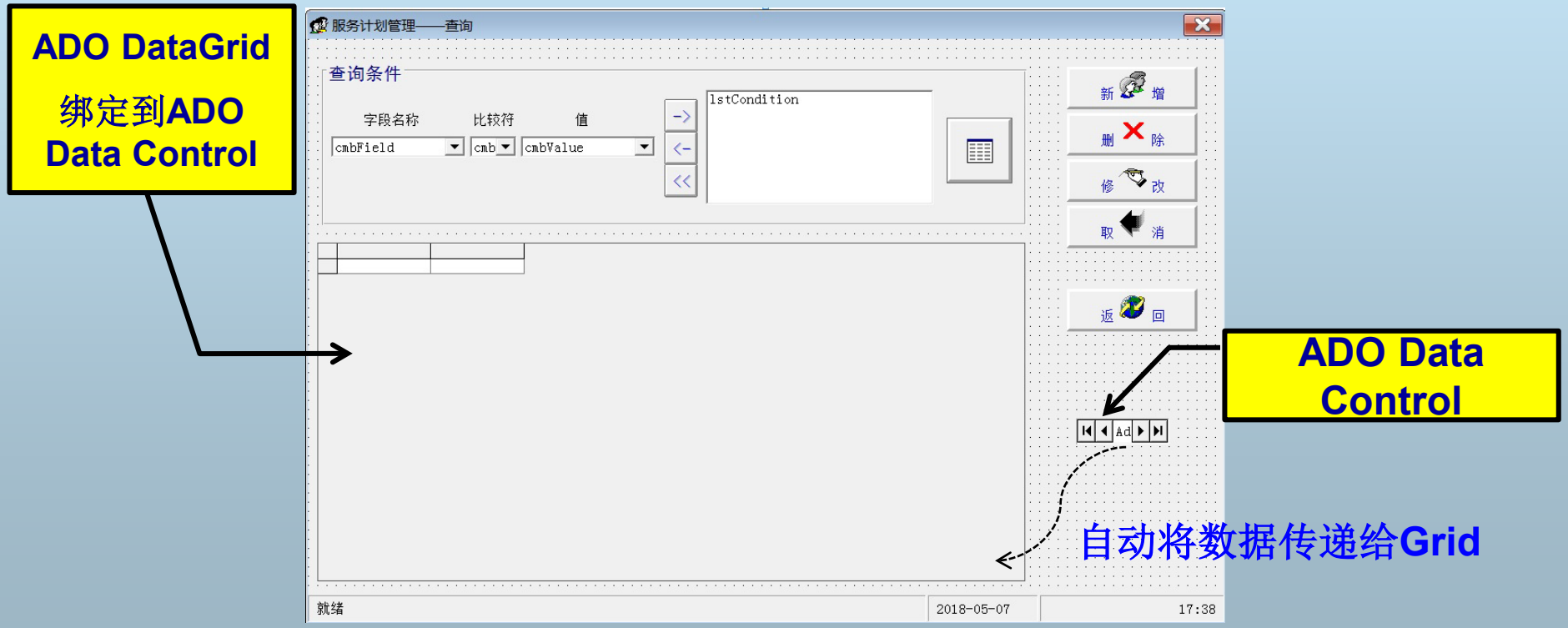
ADO数据控件

- “工程”—》“部件”
  - 选择“Microsoft ADO Data Control（OLE DB)”
- ADO Data控件代表一个数据源，如表、SQL、视图等。它可以被其它控件绑定，从而使得数据源中的数据可以自动显示在绑定控件上
- 属性
  - ConnectString
    - 用于建立连接
  - RecordSource
    - 指定数据源，可以是一个表、视图或一个SQL查询



ADO数据绑定控件

- 数据绑定控件
  - 显示ADO Data控件所连接的数据源中的数据
  - 主要属性：Datasource，指定所绑定的ADO Data Control



记录查询的实现

- 结合Recordset对象与ADO Data控件实现数据的查询
  - ADO Data控件具有一个属性Recordset，表示了一个与其关联的Recordset对象
  - 可以在程序中控制ADO Data控件的Recordset属性来改变ADO Data控件所关联的数据
  - 随后，ADO Data控件的数据绑定控件（一般是网格控件）就可以显示查询结果

记录查询的实现

- **根据输入的EmployeeID查询记录，设txtID为输入的EmployeeID，设数据网格控件dtgData用于显示结果，并与ADO Data控件adcEmployee绑定**

```
Dim strSQL as string
strSQL="select * from Employees where EmployeeID='" & txtID & "'"
adcEmployee.Recordsource=strSQL
adcEmployee.Refresh
If adcEmployee.Recordset.BOF and adcEmployee.Recordset.EOF then
    Msgbox "无匹配记录"
End If
```

ADO编程总结

- Connection对象用于创建到数据源的连接
- Command对象可用于执行某个存储过程或SQL语句
- Recordset对象可用于控制记录的增、删、改、查

本章小结

- 数据库应用系统体系结构
- 数据库应用系统开发过程
- 数据库访问编程
- ADO数据库访问示例