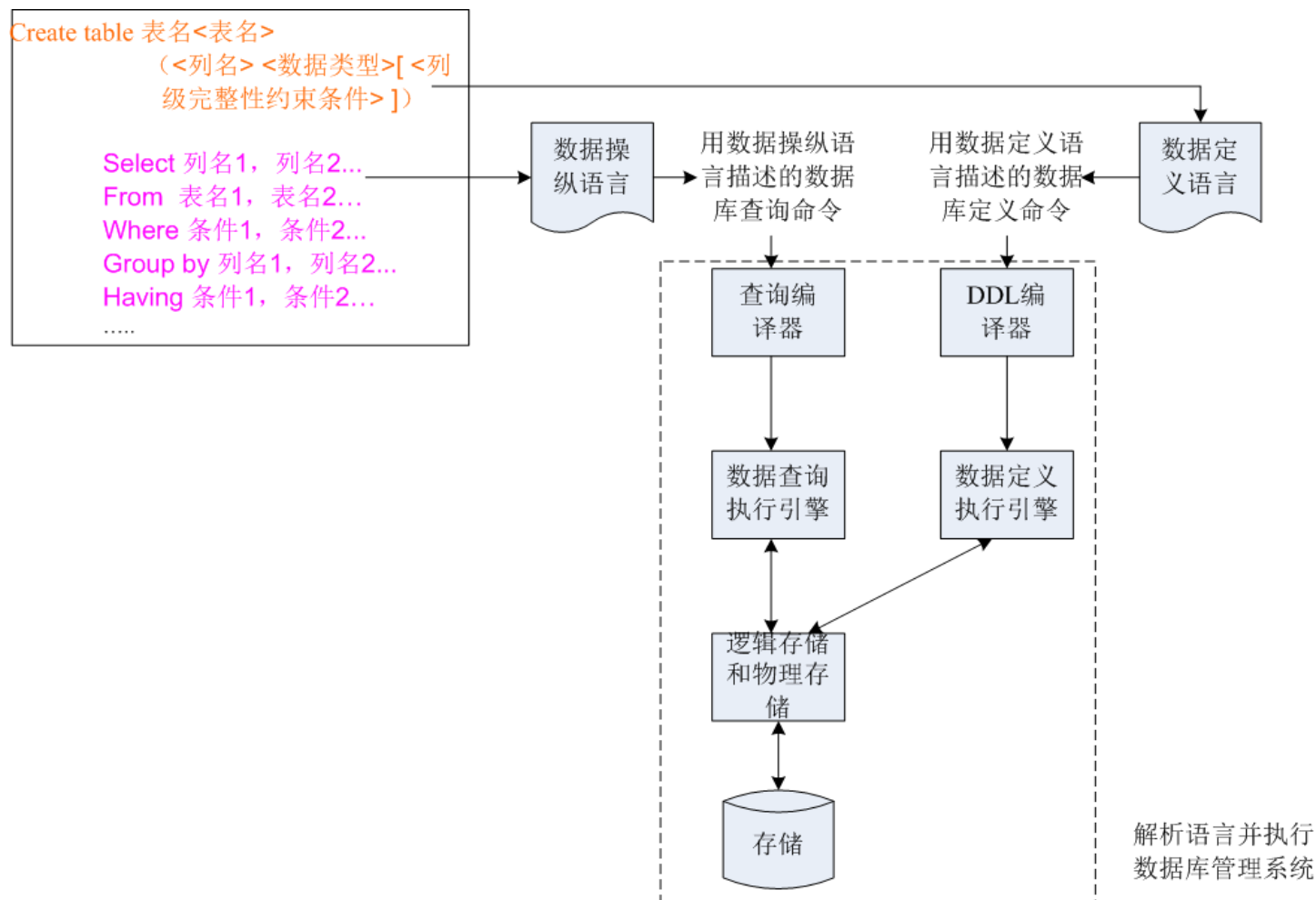


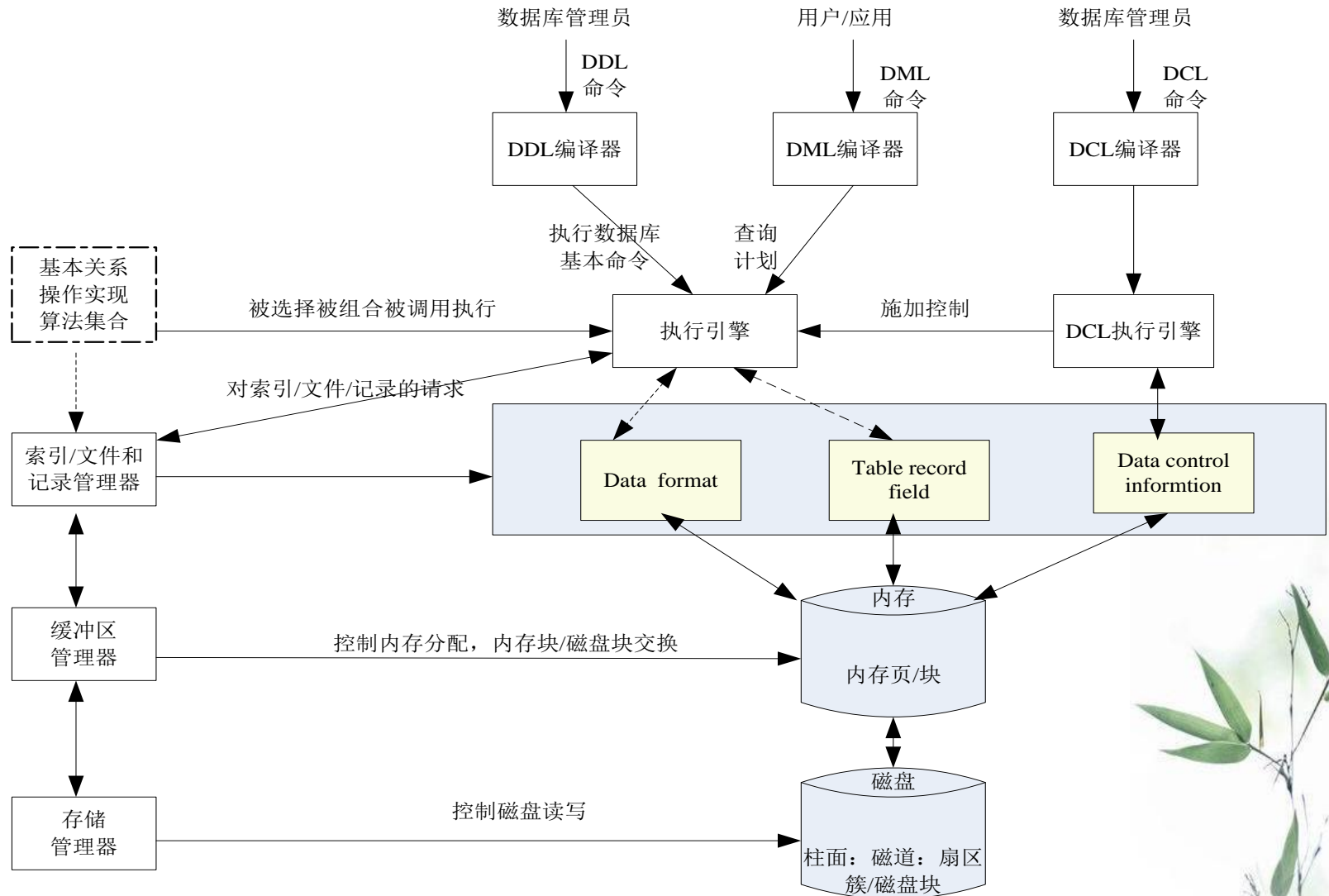
数据库系统基本概念

➤ 数据库管理系统——从系统角度看DBMS功能
形式 → 构造 → 自动化 数据库管理系统的实现



数据库系统基本概念

➤ 数据库管理系统——从系统角度看DBMS功能



数据库系统基本概念

➤数据库管理系统——从系统角度看DBMS功能

➤DBMS为完成DB管理，在后台运行着一系列程序...

□语言翻译处理：将用数据库语言书写的内容，翻译成DBMS可执行的命令。例如：DDL编译器，DML编译器，DCL编译器等；

□数据存取：提供数据在磁盘、磁带等上的高效存取手段。例如：存储管理器，缓冲区管理器，索引/文件和记录管理等；

□查询优化：提高数据库检索速度的手段；例如贯穿于数据存取各个阶段的优化程序；

□通信控制：提供网络环境下数据库操作的手段



数据库系统基本概念

➤数据库管理系统——从系统角度看DBMS功能

➤DBMS为完成DB管理，在后台运行着一系列程序...

- ❑事务管理：提供提高可靠性并避免并发操作错误的手段
- ❑故障恢复：使数据库自动恢复到故障发生前正确状态的手段
- ❑安全性控制：提供合法性检验，避免非授权非法用户访问数据库的手段
- ❑完整性控制：提供数据及数据操作正确性检查的手段
- ❑数据字典管理：管理用户已经定义的信息
- ❑应用程序接口(API)：提供应用程序使用DBMS特定功能的手段
- ❑备份、运行日志操控等实用程序
- ❑数据库数据装载、重组等实用程序
- ❑数据库性能：统计在运行过程中数据库的各种性能数据，便于优化运行
- ❑... ..



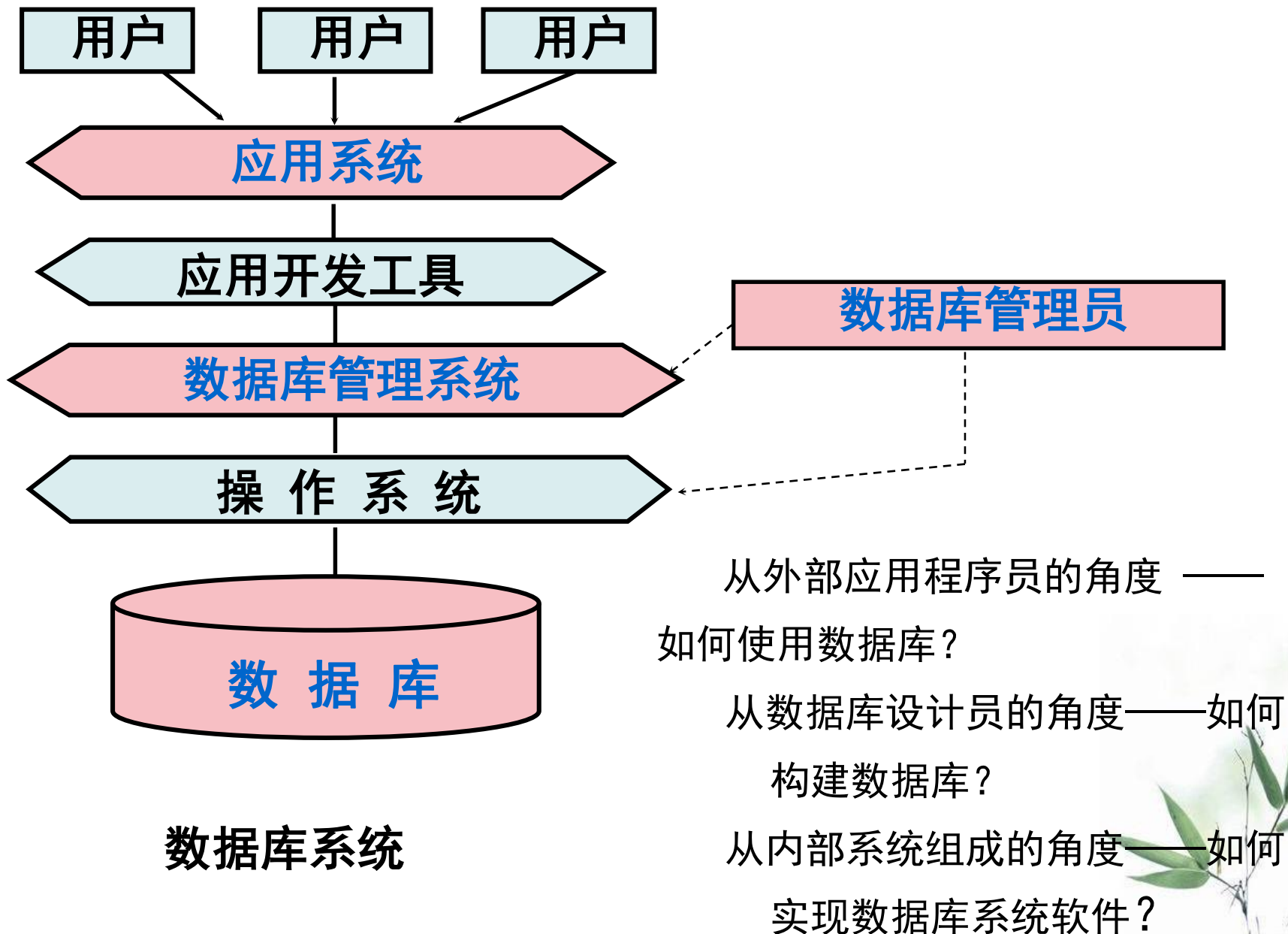
四、数据库系统

- 数据库系统（**Database System**，简称**DBS**）

在计算机系统中引入数据库后的系统构成

- 数据库系统的构成
 - 数据库
 - 数据库管理系统（及其开发工具）
 - 应用系统
 - 数据库管理员





数据库系统

1.1 数据库系统概述

1.1.1 四个基本概念

1.1.2 数据管理技术的产生和发展

1.1.3 数据库系统的特点



数据管理技术的产生和发展

- DBMS的产生动机

- 应用需求的推动
- 计算机硬件的发展
- 计算机软件的发展

推动计算机发展，DBMS出现

- 分析DBMS的产生动机

- 发现问题 找出文件系统的不足，提出DBMS
- 分析问题 模型化、建立DBMS理论
- 解决问题 设计实现DBMS



数据管理技术的产生和发展

- 数据管理技术的发展过程
 - 人工管理阶段(**20世纪40年代中--50年代中**)
 - 文件系统阶段(**20世纪50年代末--60年代中**)
 - 数据库系统阶段(**20世纪60年代末--现在**)

注：主要围绕提高数据独立性、降低数据的冗余度、提高数据共享性、提高数据的安全性和完整性等方面来进行改进，使用者能有效地管理和使用数据资源。



一、人工管理阶段

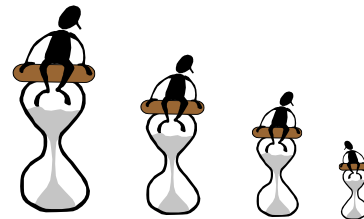
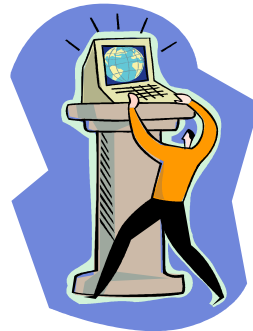
- 时期
 - 20世纪40年代中--50年代中
- 产生的背景
 - 应用需求 科学计算
 - 硬件水平 无直接存取存储设备
 - 软件水平 没有操作系统
 - 处理方式 批处理



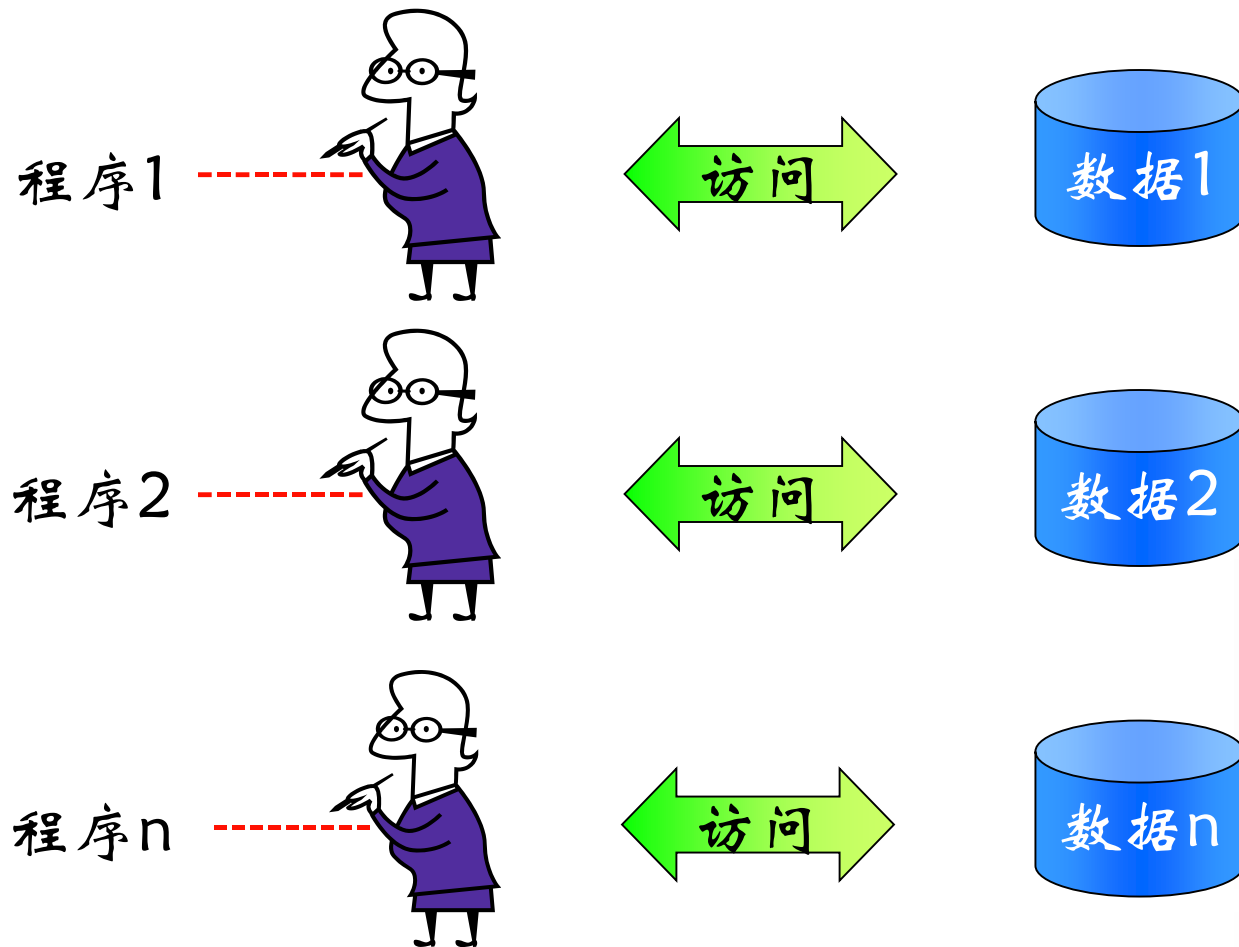
一人工管理阶段

■ 背景

- 计算机主要用于科学计算
 - 数据量小、结构简单，如高阶方程、曲线拟和等
- 外存为顺序存取设备
 - 磁带、卡片、纸带，没有磁盘等直接存取设备
- 没有操作系统，没有数据管理软件
 - 用户用机器指令编码，通过纸带机输入程序和数据，程序运行完毕后，由用户取走纸带和运算结果，再让下一用户上机操作



应用程序与数据的对应关系(人工管理阶段)



应用程序与数据的对应关系(人工管理阶段)

对学生成绩进行排序的C程序：

```
#include <stdio.h>
#define N 10
int main()
{
    int x[N]={73,65,92,83,72,64,82,63,58,94};
    int i,j,k,t;
    for(i=0;i<N-1;i++)
    {
        for(j=0;j<N-1-i;j++)
            if (x[j]>x[j+1])
            {
                t=x[j];
                x[j]=x[j+1];
                x[j+1]=t;
            }
    }
    printf("The sorted numbers:\n");
    for(i=0;i<N;i++)
        printf("%5d",x[i]);
    printf("\n");
    return 0;
}
```

• 特点

- 数据的管理者：用户（程序员），数据不保存
- 数据面向的对象：某一应用程序
- 数据的共享程度：无共享、冗余度极大
- 数据的独立性：不独立，完全依赖于程序
- 数据的结构化：无结构
- 数据控制能力：应用程序自己控制



二、文件系统阶段

- 时期

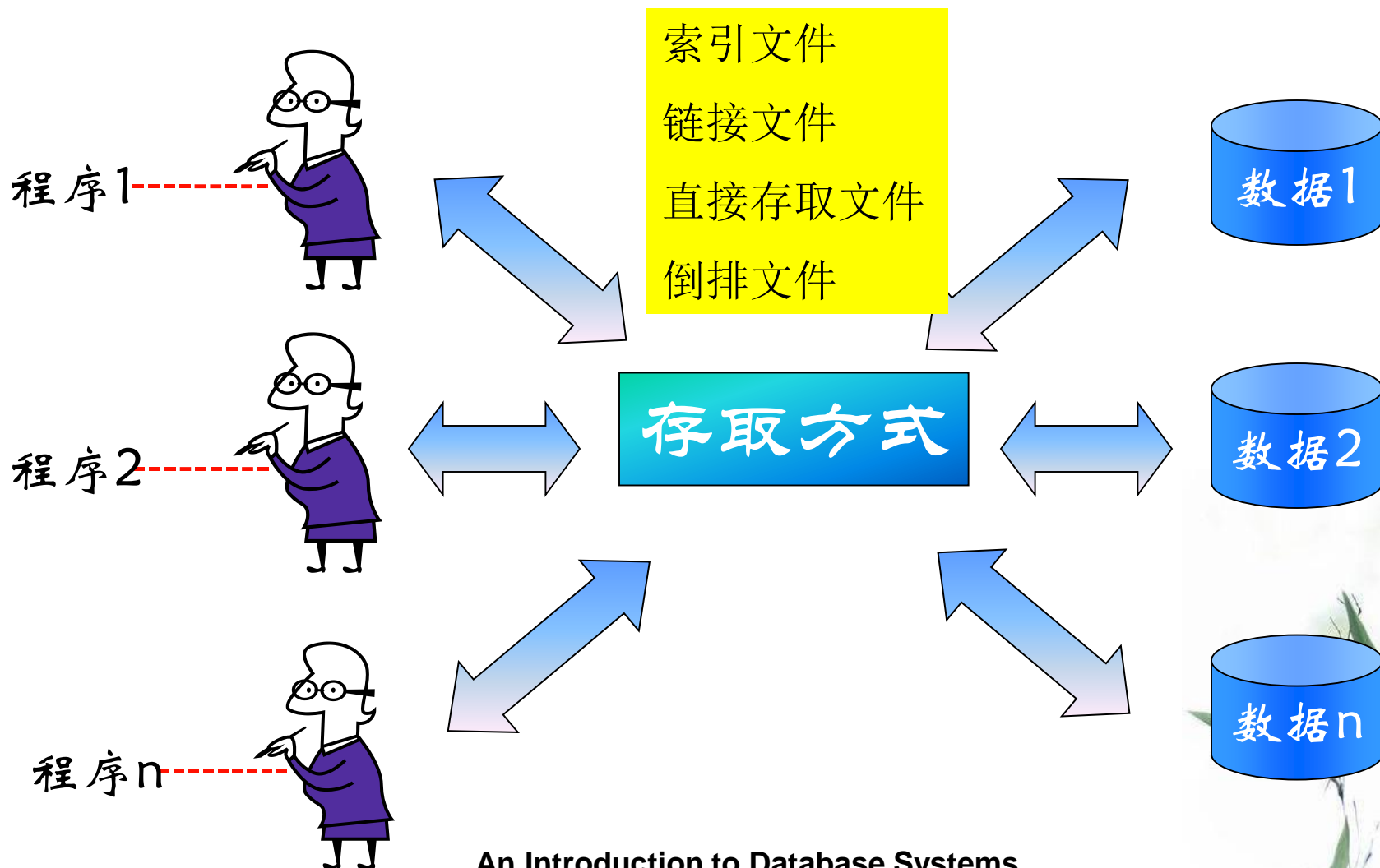
- 20世纪50年代末--60年代中

- 产生的背景

- 应用需求 科学计算、管理
 - 硬件水平 磁盘、磁鼓
 - 软件水平 有文件系统
 - 处理方式 联机实时处理、批处理



应用程序与数据的对应关系(文件系统阶段)



应用程序与数据的对应关系(文件系统阶段)

对学生成绩进行排序的C程序:

```
#include <stdio.h>
#define N 10
int main()
{
    int x[N],i,j,k,t;
    FILE *fp;
    fp=fopen("e:\\score.dat","r"); //打开文件
    for(i=0;i<N;i++)
        fscanf(fp,"%d",x+i); //从文件读数据
    for(i=0;i<N-1;i++)
    {
        for(j=0;j<N-1-i;j++)
            if (x[j]>x[j+1])
            {
                t=x[j];
                x[j]=x[j+1];
                x[j+1]=t;
            }
    }
    printf("The sorted numbers:\n");
    for(i=0;i<N;i++)
        printf("%5d",x[i]);
    printf("\n");
    fclose(fp); //关闭文件
    return 0;
}
```

❖ 特点

数据的管理者：文件系统，数据可长期保存

数据面向的对象：某一应用程序

数据的共享程度：共享性差、冗余度大

数据的结构化：记录内有结构，整体无结构

数据的独立性：独立性差，数据的逻辑结构

改变必须修改应用程序

数据控制能力：应用程序自己控制



—文件系统阶段



劳资科

学号	姓名	系别	补贴
----	----	----	----



房产科

学号	姓名	性别	系别	住址
----	----	----	----	----



学籍科

学号	姓名	系别	学分	学位
----	----	----	----	----



人事科

学号	姓名	性别	系别	年龄	学位	出身
----	----	----	----	----	----	----



思考一下

在数据库系统还没有出现的时候，如何实现一个学校管理系统？

1. 一个完成数据输入、存储、修改、输出、持久有效的系统
2. 存储数据 —— 使用一种数据结构
3. 易于修改数据 —— 这种数据结构便于查找数据和修改数据
4. 持久有效 —— 系统有安全保护措施



- 用 文件系统开发管理应用

- 开发任务

- 举例：完成学校管理系统模块之一：学籍管理模块，包括学生注册、选课、学籍和成绩管理。

- 开发工具及环境

- 程序设计语言如C/C++，Windows下的文件系统。

- 数据结构设计

- 确定管理对象；抽取对象主要特征；将特征组织起来。

- 利用C/C++语言中的“结构”数据类型存放数据。



文件系统实现过程（1） - 数据结构定义

struct **Student**

```
{    int    nStudNo;  
    char    szStudName[20];  
    char    cGender;  
    int     nAge;  
    char    szDept[30];  
};
```

struct **Enrollment**

```
{    int    nStudNo;  
    int    nWhichTerm;  
    char    cEnrolled;  
    char    szMem[30];  
};
```

struct **Course**

```
{ int    nCourseNo;  
  char    szCourseName[20];  
  char    szDept[30];  
};
```

struct **Score**

```
{    int    nStudNo;  
    int    nCourseNo;  
    int    nScore;  
};
```

注：sz - 数组（Array），n - 整数（Number），c - 字符（Char）

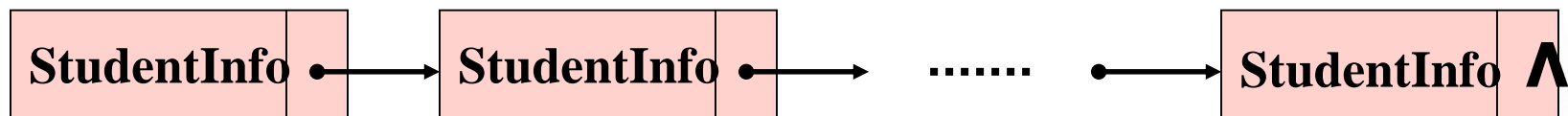
文件系统实现过程（2）

•构造链表

为了管理学生的信息，常常以班级为单位。

如果对集合数据进行添加、修改、排序等操作，一个办法是构造链表。在我们的例子中，至少需要构造四个这样的链表。

以学生基本信息链表为例：



•链表操作

对班级的学生进行增加、删除、修改、查询与排序工作。

相对于链表是对结点进行insert, delete, update, query, sort.



文件系统实现过程 (3) - 用户界面设计



学生学籍管理

学号 20030900001

姓名 张三

性别 M

年龄 18

所属系 计算机

增加 删除 修改 查询 退出

学生学籍管理界面

文件系统实现过程（4）

•创建数据存储文件

将链表中的数据以文件形式存盘。使用文件I/O操作函数完成对文件中数据的读写,上例中需要四个数据文件。

程序运行前,需将数据文件中的数据读出,放入程序中的对应链表中,以方便数据的操作。

程序结束运行之前,将链表中的数据存入到对应的数据文件中。

•数据管理操作

最基本的数据操作：增加、删除、修改和查询，简称：**增删改查**。其他操作或功能由这四个基本操作组合而来



文件系统的缺陷（1）

- 无法处理超大量数据（如GB级，TB级，PB级）
 - 证券、银行、保险等领域拥有超大量级数据。
 - 内存不够，不能一次读入大量数据。
 - 大数据量下的查询速度存在问题
- 多用户并发访问（Concurrent Access）
 - 前面的学生管理系统只能单机或单用户使用。
 - 多用户的并发访问可能导致多个用户同时存取同一数据（Conflict）。
 - 数据访问冲突导致数据不一致。
 - 多用户并发访问的透明度。



文件系统的缺陷（2）

- 故障情况下的恢复（Crash Recovery）
 - 文件系统需要增加大量代码
- 安全性（Security）
 - 文件系统必需做很大改进，以满足数据安全的要求，例如：不同用户的授权问题。
- 数据的完整性（Integrity）
 - 同一数据可能会出现在多个数据结构中，同时对应地出现在多个数据文件中。
 - 修改一个文件的一个数据，必须同时修改其他文件中的相同的数据，否则就会产生数据的不一致。



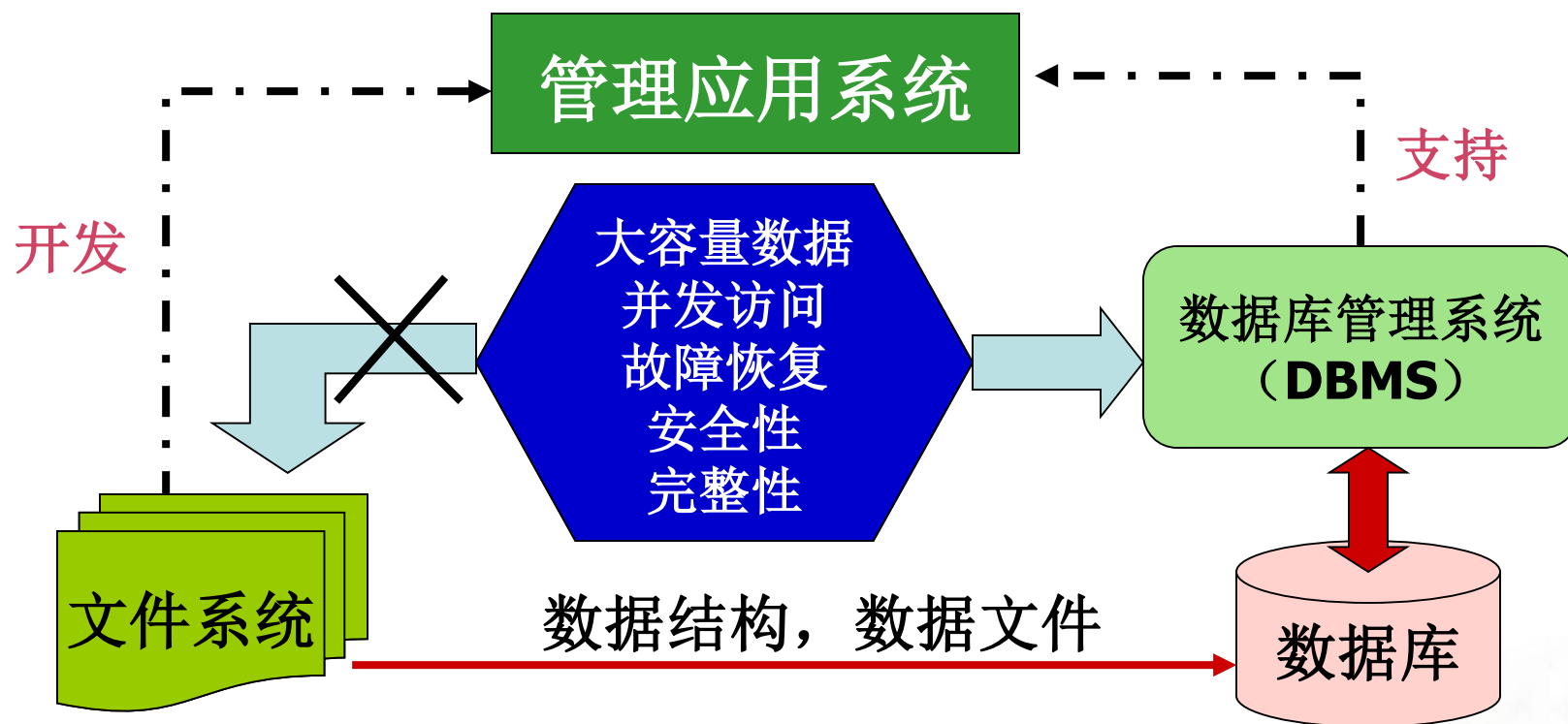
如何解决文件系统缺陷

- 解决方案

- 开发一种比文件系统更加强大的数据管理系统，将数据的**结构语义、关联关系、冗余消除**等交给该系统管理，并提供**一次一集合**的数据访问方式。
- 使用资源抽象、资源共享等**虚拟技术**，自动保证数据并发共享访问的安全性、隔离性。每个应用对各自的虚拟数据库操作，系统保证虚拟资源到物理资源的**映射**，保证物理数据库的**一致性、完整性**
- 能自动解决**共享异常、原子性、完整性**问题



数据库诞生



- 抽出五个基本公共功能形成数据库管理系统
- 引起的术语变化:
 - 结构类型 \Rightarrow 数据模型 (data model)
 - 具体的某个结构 \Rightarrow 数据模式 (data schema)
 - 数据文件 \Rightarrow 数据库

三、数据库系统阶段

- 时期
 - 20世纪60年代末以来
- 产生的背景
 - 应用背景 大规模管理
 - 硬件背景 大容量磁盘、磁盘阵列
 - 软件背景 有数据库管理系统
 - 处理方式 联机实时处理,分布处理,批处理



1.1.3 数据库系统的特点

- 数据结构化
- 数据的共享性高，冗余度低，易扩充
- 数据独立性高
- 数据由**DBMS**统一管理和控制

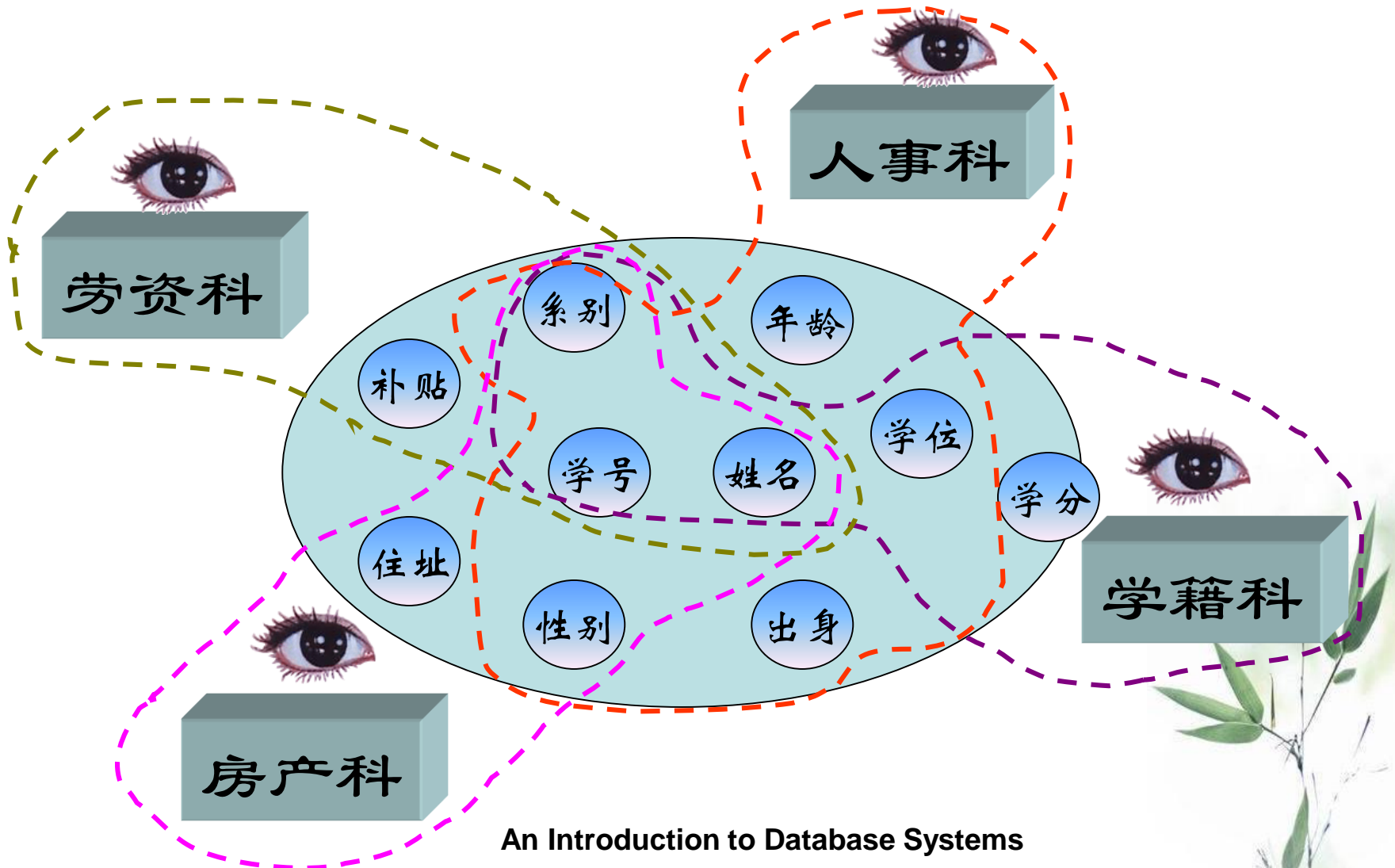


数据结构化

- 整体数据的结构化是数据库的主要特征之一
- 数据库中实现的是数据的真正结构化
 - 数据的结构用数据模型描述，无需程序定义和解释
 - 整体是结构化的，数据之间具有联系
 - 数据的最小存取单位是数据项



—数据库系统阶段



数据的共享性高，冗余度低，易扩充

- 数据库系统从整体角度看待和描述数据，数据面向整个系统，可以被多个用户、多个应用共享使用
- 数据共享的好处
 - 减少数据冗余，节约存储空间
 - 避免数据之间的不相容性与不一致性
 - 使系统易于扩充



数据独立性高

- 物理独立性
 - 指用户的应用程序与存储在磁盘上的数据库中数据是相互独立的。
当数据的物理存储改变了，应用程序不用改变。
- 逻辑独立性
 - 指用户的应用程序与数据库的逻辑结构是相互独立的。数据的逻辑结构改变了，用户程序也可以不变。
- 数据独立性是由DBMS的二级映像功能来保证的



数据由DBMS统一管理和控制

DBMS提供的数据库控制功能

– (1)数据的安全性（Security）保护

保护数据，以防止不合法的使用造成的数据的泄密和破坏。

措施：用户标识与鉴定，存取控制，视图，数据加密等

– (2)数据的完整性（Integrity）检查

将数据控制在有效的范围内，或保证数据之间满足一定的关系。

措施：完整性约束条件定义和检查



数据由DBMS统一管理和控制

- DBMS提供的数据库控制功能

- (3)并发（Concurrency）控制

对多用户的并发操作加以控制和协调，防止相互干扰而得到错误的结果。

措施：封锁

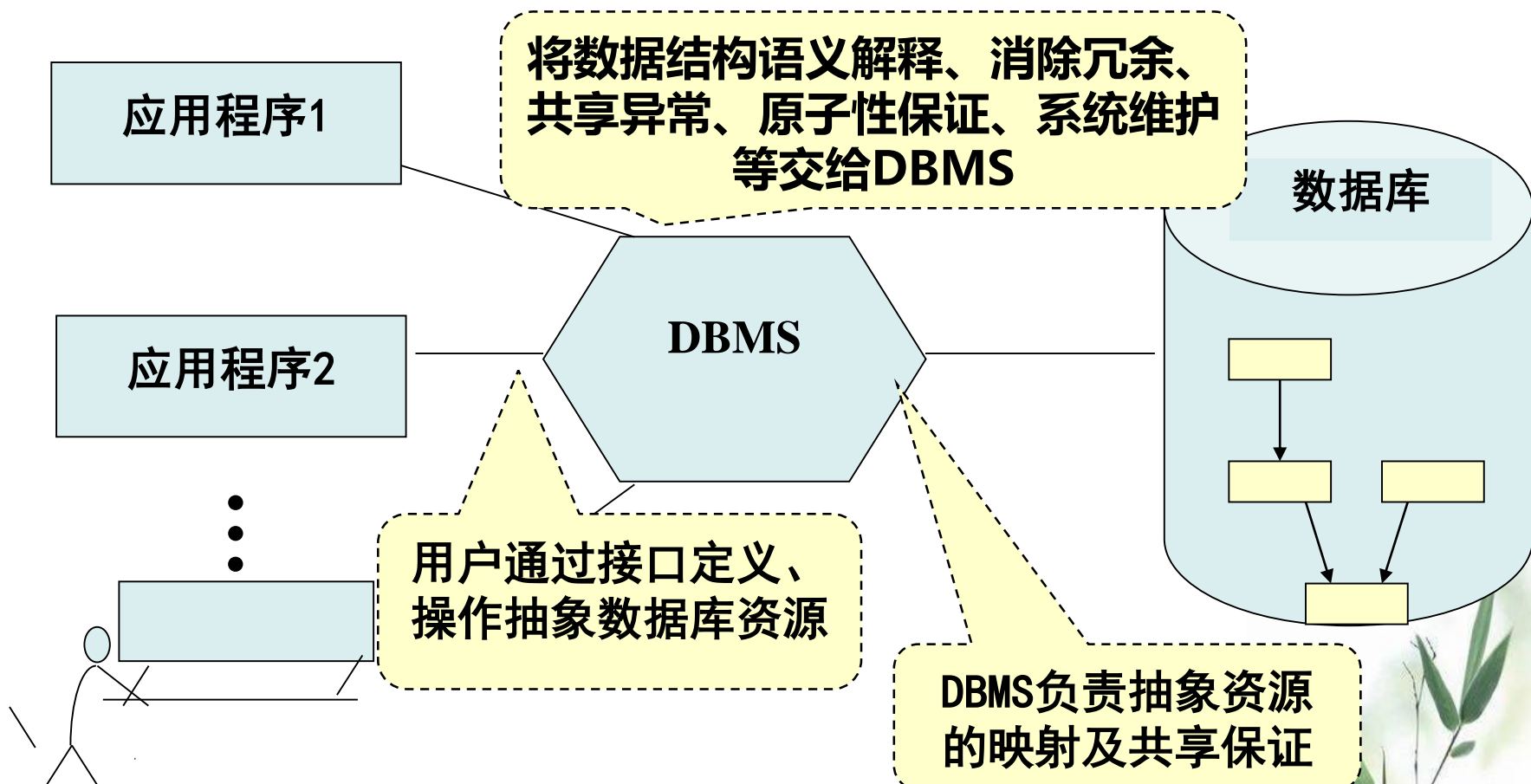
- (4)数据库恢复（Recovery）

将数据库从错误状态恢复到某一已知的正确状态。

措施：转储，镜像，日志



应用程序与数据的对应关系



数据库系统阶段应用程序与数据之间的对应关系