



第四章 数据库管理系统引论



4.1 DBMS结构简介

数据库管理系统 (DBMS) 是数据库系统的核心，它对数据库系统的功能和性能有决定性影响。

DBMS最基本的功能是正确、安全、可靠地执行数据库语言语句。图4-1表示一个解释执行的关系DBMS的结构，可以从中了解DBMS的一般工作原理和主要组成部分。

与高级程序设计语言一样，DBMS有两种实现方法——编译和解释。

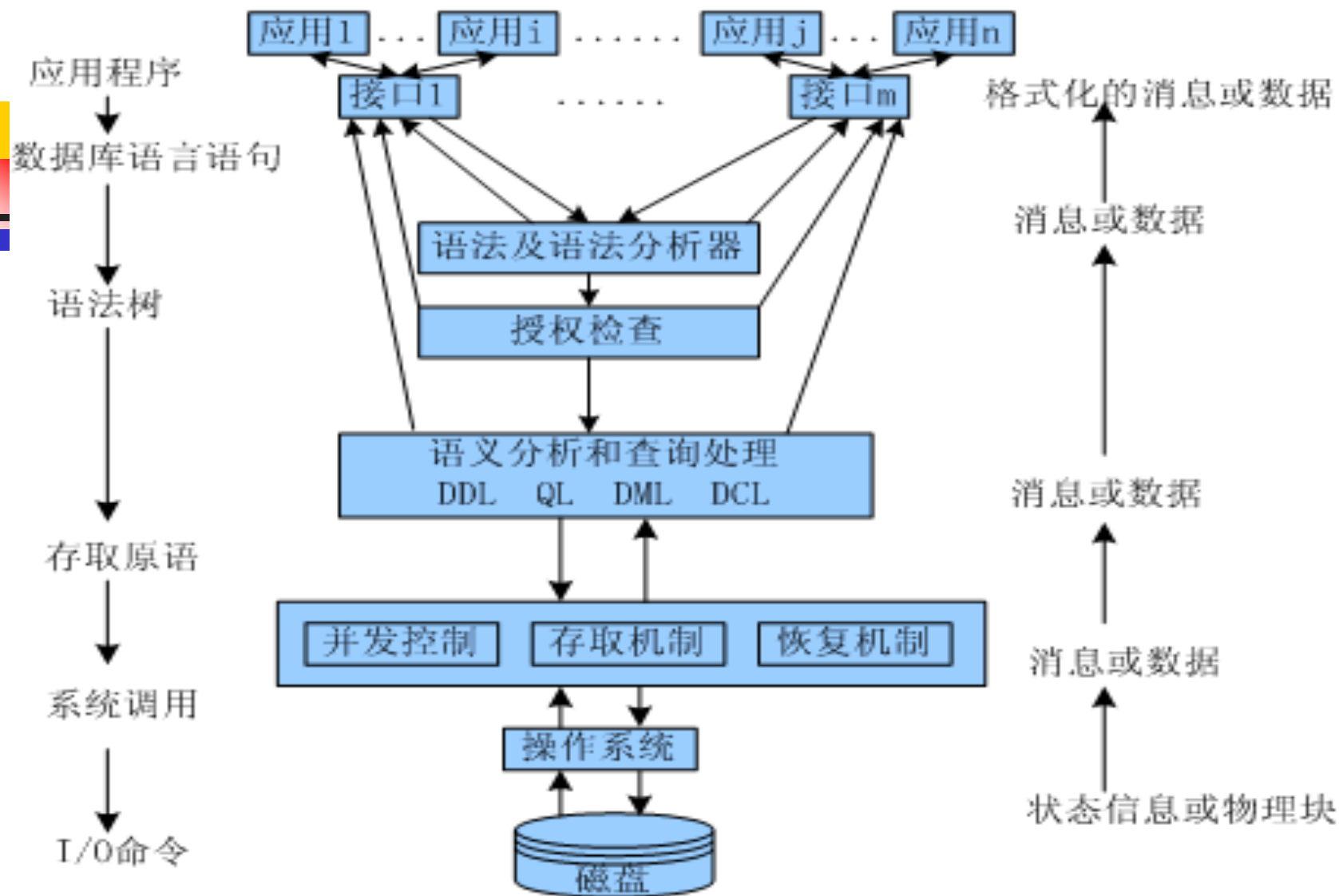


图4—1 DBMS结构



4.2 事务 (transaction)

- 事务是DBMS的执行单位, 由有限的数据库操作序列组成, 一般要求事务具备下列性质:

1. 执行的原子性 (Atomic)

事务执行时应遵守 “要么不做, 要么全做” (nothing or all) 的原则。



2.功能上的一致性（Consistency）

事务的作用应使数据库由一个**一致状态**转变到另一个一致状态。

3.彼此的隔离性（Isolation）

如果多个事务并发执行，应像各个事务独立执行一样。——由“**并发控制**”保证。



4.作用的持久性（Durability）

一个成功执行的事务对DB的影响应是持久的，即使DB因故障受到破坏，也应能恢复。

这四个性质称为**事务的ACID准则**。

下面是一个事务的例子，它将款项由A账户拨给B账户。

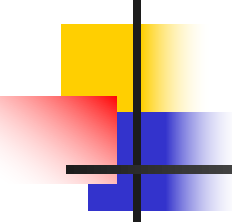


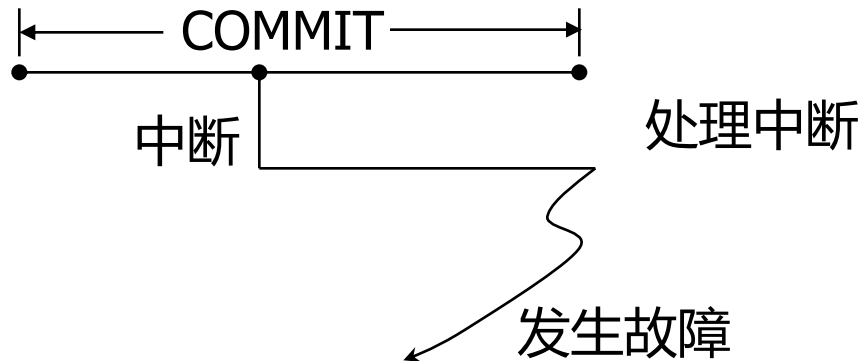
示例

```
BEGIN TRAN
  read A
  A ← A - S
  if A < 0 then      /* A款不足*/
    begin
      display “A款不足”
      ROLLBACK /*出口1*/
    end
  else
    begin
      B ← B + S
      display “拨款完成”
      COMMIT /*出口2*/
    end
  end
```

ROLLBACK 撤销事务的影响，相当于 “do nothing”

**COMMIT 提交，相当于 “do all”。
只有在COMMIT之后，事务对数据库产生的变化才对其它事务开放。
(为什么?)**

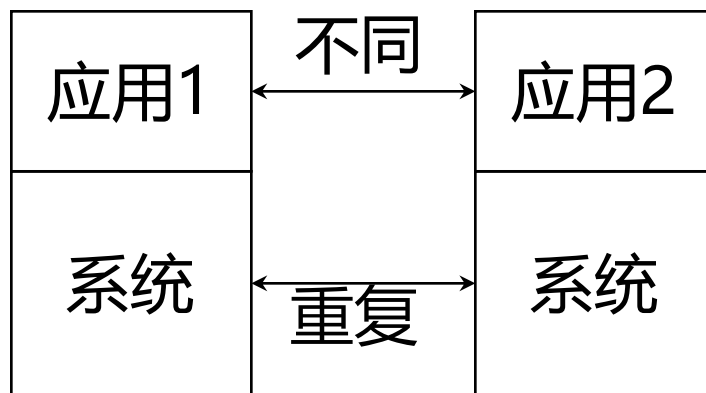
- 
- 事务的出口：commit 或rollback
 - 只有在执行commit之后，事务对数据库所产生的变化才对其他事务开放。
 - 执行commit命令时，要封闭中断，以防处理中断时发生故障



4.3 DBMS的进程结构

- DBMS进程结构的划分主要着眼于结构合理和性能提高。
- 应用进程，系统进程（可重入）

不划分：





目前，多数DBMS把主要功能组成一个DBMS**核心进程**，也有些DBMS除了核心进程外，还把一些可以“缓办”的公共操作组成几个后台服务进程。

例如预读取可能用到的物理块，延迟写入缓存中的内容，网络服务管理，撤销事务，清除异常结束的DBMS进程等。这些进程在DBMS启动时就建立，为各个事务服务。



下面主要考虑DBMS核心进程的结构方案：

1.一个应用进程对应一个DBMS核心进程

优点：实现容易

缺点：

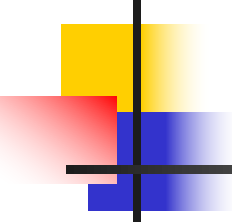
- (1).进程的创建、撤销、通信和切换的开销大。**
- (2).并发事务的增加，进程数激增，内存空间有限，性能下降。**
- (3).不利于事务共享内存空间。**



2.单进程多线程DBMS进程结构

线程是现代OS引入的概念。

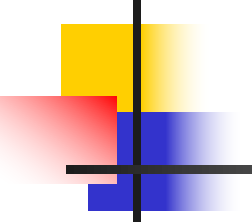
- 以线程为程序并发执行的单位；
- 一个进程中可创建多个可以相互切换的线程；
- 这些线程中至少有一个处于就绪状态，进程才处于就绪状态；
- 进程运行时，其中必有一个线程运行；
- 同一进程所属的线程共享进程占用的资源，属于线程本身的专用资源很少，描述线程的状态也比进程要少，因此，线程所需资源比进程少；
- 线程的切换开销和线程间的通信开销小。



在多处理机系统中，引入线程，增强了进程的可并发程度。

单进程多线程的DBMS中，系统只创建一个DBMS进程（用户接口仍然是进程）。该进程中有常驻的公共服务线程和应用户要求而创建的用户线程。

——DBMS的并发执行从进程级改为线程级。



尽管很多现代OS的核心具有线程管理的功能，但对DBMS来说，还是在DBMS进程（**相对于OS，是用户进程**）中实现线程为宜。理由如下：

- （1）可以按照DBMS的需要确定线程调度策略；
- （2）线程的切换在用户态，不必转入操作系统的核心态，切换开销小；
- （3）可以在不支持线程的操作系统上运行，减少对操作系统的依赖，有利于提高操作系统的可移植性。



由DBMS管理线程，需要OS提供如下支持：

(1) .提供**非阻塞I/O** (Nonblocking I/O) 和**异步I/O** (asynchronous I/O) 功能；

(2) .支持 “**公平**” 调度 (fair schedule) ；

即不把具有多线程的DBMS进程，与其它进程等同看待，应区分轻重。



4.4 DBMS的系统结构

1.分时系统环境下的**集中式**数据库系统结构

应用的要求以及软硬件条件决定了数据库系统以集中为宜，数据库建立在本单位的主要计算机上，用户通过终端或远距离终端分时访问。

数据及其管理都是集中的，数据库系统的所有功能，从用户接口到DBMS核心都集中在DBMS所在的计算机上。

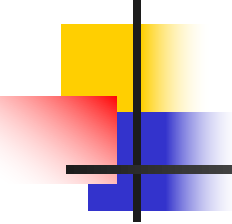


2.网络环境下的**客户/服务器**结构

20世纪70年代：微机的出现和迅速发展；计算机网络的发展和广泛应用，改变了计算机应用系统的格局。

客户机/服务器是一种特殊的分布式处理系统。其中，有一至多台称为客户机的计算机和一至多台称为服务器的计算机通过网络联接。

可以将DBMS的核心部分放在服务器中，而客户机处理数据库的接口部分。客户机也可以有自己的局部DBMS。



客户机面向用户，接受任务，并将任务中需要由服务器完成的部分委托服务器执行。而服务器只接受客户机的委托，完成特定的任务，例如数据库服务。因此，处理是分布的，数据却是集中的，仍属于集中式数据库系统。



问题1：

网络环境下的打印服务器、文件服务器属于客户/服务器结构吗？

不属于，打印服务器、文件服务器的处理仍然是集中的。



问题2:

如果有多个数据库服务器呢？还属于集中式数据库系统吗？

即使系统中有多个数据库服务器，也只是多个集中的数据库，这些库中的数据彼此独立，其联系只能由应用程序自己解决。



客户器与服务器划分界面的一般原则是：

- (1) 客户提供用户接口、执行应用程序，对服务器提出服务请求；**
- (2) 服务器只完成客户器委托的公共服务；**
- (3) 服务器与客户器间的数据交换量要尽可能的少；**



例如，MS SQL Server, Oracle

三层结构：



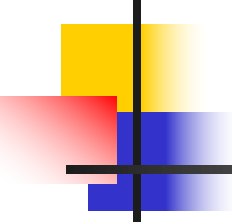


3.物理上分布、逻辑上集中的**分布式**数据库结构

数据共享和数据集中管理是数据库的主要特征。随着单位规模的扩大和地理上的分散，集中式数据库系统有如下缺点：

- 通信开销大
- 性能差，瓶颈
- 可用性差
- 可扩充性差
- 难以管理

由于存在这些缺点，从20世纪70年代后期，开始了分布式数据库系统的研究。



物理上分布、逻辑上集中的分布式数据库结构的思想是：把全局数据模式按数据的来源和用途，合理分布在系统的多个节点上，使大部分的数据可以就近存取。

逻辑上，用户看到的是一个数据模式为全局数据模式的集中式数据库。

缺点：全局数据模式很难设计、管理、扩充和修改（类似高度集中的计划经济难以管理）。



4.物理上分布、逻辑上分布的分布式数据库结构

(事实上, 对大范围统一的逻辑几乎不可能)

特点:

- (1) 节点自治**
- (2) 没有全局数据模式**

每个节点看到的数据模式:

- (1) 本节点的数据模式**
- (2) 供本节点共享的其它节点上有关的数据模式**

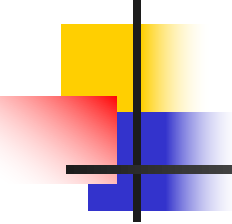


没有全局数据模式，节点数据模式的修改甚至节点的加入、撤离，仅仅影响有关的节点。
这种分布式数据库系统又称为“联邦式数据库系统**” (federated distributed database system) 。**



4.5 数据目录

- **数据目录 (catalog) 存放一组关于数据的数据 (描述数据模式的数据) , 也叫元数据 (meta-data) 。**
- **DBMS的任务是管理大量的、共享的、持久的数据, 有关这些数据的描述需长期保存, 一般把这些元数据组成若干表, 即数据目录。**

- 
- **数据目录既是数据，又不同于一般数据，**
数据目录也是表，可供查询，主要为DBMS服务，数据目录本身的定义和描述也包含在数据目录中。数据目录只能由系统定义，为系统所有。在初始化时，由系统自动生成（递归初始，类比编译的符号表）

数据目录中一般包含下列表：
SYSTAB、SYSCOL、SYSIDX、SYSVIEW、
SYSVWATR



元数据可以分为2类：

(1) 相对稳定：基表、视图和索引的定义；

(2) 经常变化：数据库状态的统计，例如，元组个数、现有不同属性值的个数等——主要用于查询优化，不必太准，可以定期更新。

数据目录是影响系统全局的以读为主的数据，对系统的效率影响很大。