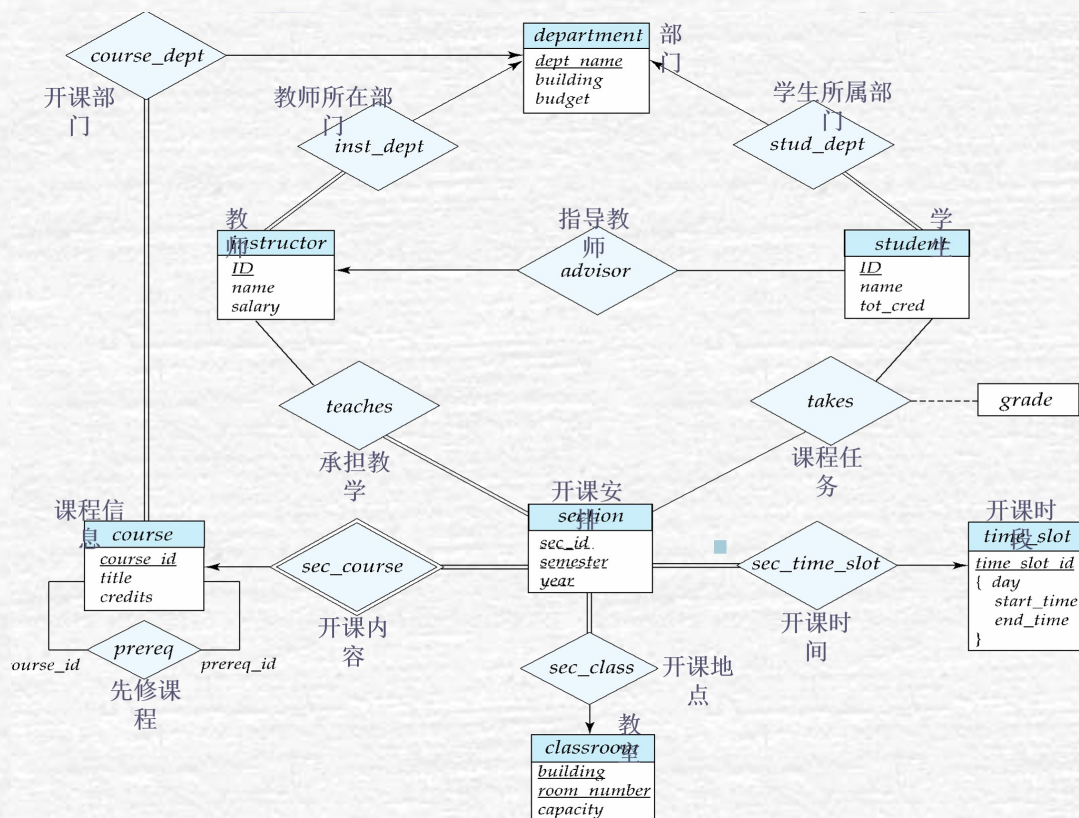




# 物理设计(数据库存储技术)

单 位：重庆大学计算机学院

# 数据库设计主要有哪些环节?



course_id	title	dept_name	credits
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3
EE-181	Intro. to Digital Systems	Elec. Eng.	3
FIN-201	Investment Banking	Finance	3
HIS-351	World History	History	3
MU-199	Music Video Production	Music	3
PHY-101	Physical Principles	Physics	4

图 2-2 course 关系

ID	name	dept_name	salary
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

图 2-1 instructor 关系

- 数据库逻辑设计后得到什么?
- 二维表如何存储呢?

# 主要学习目标

- 表记录的存储方式。
- 表结构的存储方式。
- 在进行数据库设计之后，能够理解在关系型数据库中二维表数据的存储方式以及表结构的存储方式。



# 一 文件组织<sub>p. 255</sub>

讨论:如何在物理  
存储介质上组织  
数据?

**物理设计任务：**考虑用**文件**表示逻辑数据模型(**数据库模式**)的不同**方式**。

一个**数据库**被映射到多个不同的**文件**(file)，文件由**操作系统**来管理，这些文件被永久存储在**磁盘**上！

一个**文件**在逻辑上被组织成**记录**的一个**序列**，记录被映射到**磁盘块**(block)上。

每个文件(file)被分成定长的存储单元-**块**(block)，块是数据**存储和传输的基本单位**(默认一般是4-8KB)。

一个块可以包括**很多记录**(假设一个记录总比块小；对大数据如图片需单独处理和存储)，且一个记录的数据**不能跨块**存储

# 文件组织

```
type instructor = record
  ID varchar(5);
  name varchar(20);
  dept_name varchar(20);
  salary numeric(8, 2);
end
```

- 为每条记录分配固定大小存储空间
- 可以简单地在连续空间中依次存储记录（类似线性表）
  - 记录不能跨块存储，因此每一块中只分配能完整容纳的最大记录数，会造成少量空间浪费
- 删除记录
  - 移动其他记录
  - 空闲列表组织被删除记录空间，进行再利用

## 定长记录

记录0	10101	Srinivasan	Comp. Sci.	65000
记录1	12121	Wu	Finance	90000
记录2	15151	Mozart	Music	40000
记录3	22222	Einstein	Physics	95000
记录4	32343	El Said	History	60000
记录5	33456	Gold	Physics	87000
记录6	45565	Katz	Comp. Sci.	75000
记录7	58583	Califieri	History	62000
记录8	76543	Singh	Finance	80000
记录9	76766	Crick	Biology	72000
记录10	83821	Brandt	Comp. Sci.	92000
记录11	98345	Kim	Elec. Eng.	80000

图 10-4 包含 account 记录的文件

每条记录分配可容纳的最大字节数（53字节）

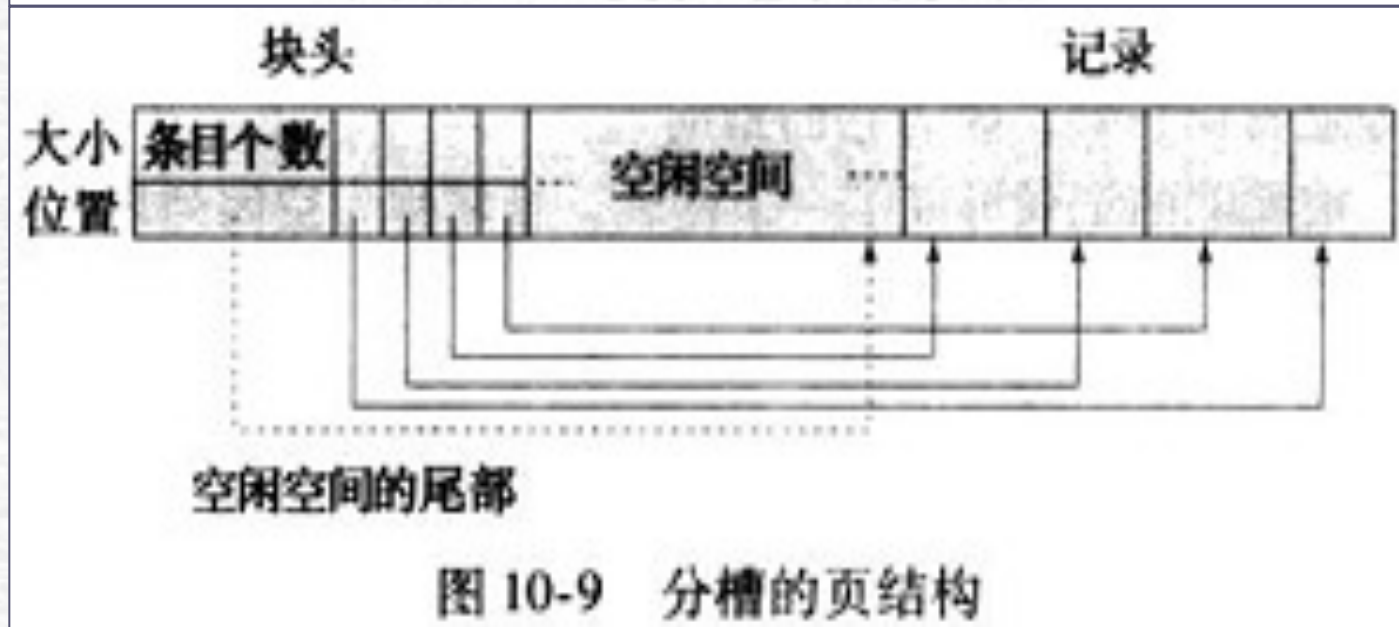
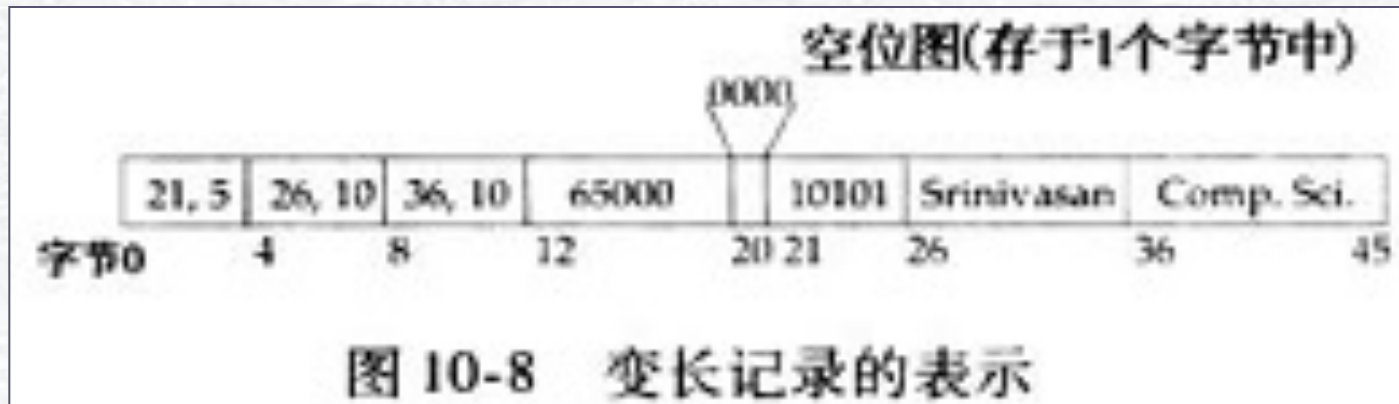
## 变长记录

- 数据库中变长记录出现的方式：
  - 一个文件中存储多种类型的记录
  - 记录中有变长字段
  - 记录中有可重复字段，如数组
- 实现变长记录，要解决的问题：
  - 如何表示记录，使得能访问每个字段
  - 如何在物理块中存储变长记录，使得能访问每条记录



# 一 文件组织

## 变长记录



## 二 (数据库)文件中记录的组织

讨论：如何在物理存储介质上组织数据库数据？

- 文件中记录的组织方式
  - 堆文件：记录在文件空间中任意放置
  - 顺序文件：按一定的顺序在文件中组织记录
  - 散列文件：按照散列函数计算值存放相应记录
  - 多表簇集文件：不同关系表里的记录存放在同一个文件中。



## 二 (数据库)文件中记录的组织

### 1. 顺序文件组织<sub>p. 259</sub>

1) 顺序文件在逻辑上是如何组织数据库数据(关系)的?

顺序文件的逻辑组织方式:

- 1) 将关系中记录按“某属性/组-**搜索码**”排列
- 2) 并用指针将记录依序连接

**特点:** 按搜索码搜索的效率高

顺序文件的物理组织方式:

- 1) 将关系中记录按**搜索码**次序进行物理存储
- 2) 采用**定长记录**或**变长记录**方式
- 3) 一个记录的信息不能分存在两个物理块中

10101	Srinivasan	Comp. Sci.	65000	
12121	Wu	Finance	90000	
15151	Mozart	Music	40000	
22222	Einstein	Physics	95000	
32343	El Said	History	60000	
33456	Gold	Physics	87000	
45565	Katz	Comp. Sci.	75000	
58583	Califieri	History	62000	
76543	Singh	Finance	80000	
76766	Crick	Biology	72000	
83821	Brandt	Comp. Sci.	92000	
98345	Kim	Elec. Eng.	80000	

图 10-10 instructor 记录组成的顺序文件

2) 顺序文件的物理存储如何实现?

## 二 (数据库)文件中记录的组织

### 1. 顺序文件组织<sub>p. 259</sub>

改善方法(指针管理)

删除和插入记录时的开销大(大量移动记录)!

10101	Srinivasan	Comp. Sci.	65000	
12121	Wu	Finance	90000	
15151	Mozart	Music	40000	
22222	Einstein	Physics	95000	
32343	El Said	History	60000	
33456	Gold	Physics	87000	
45565	Katz	Comp. Sci.	75000	
58583	Califieri	History	62000	
76543	Singh	Finance	80000	
76766	Crick	Biology	72000	
83821	Brandt	Comp. Sci.	92000	
98345	Kim	Elec. Eng.	80000	

图 10-10 instructor 记录组成的顺序文件

当待插入位置没有空间时, 用溢出块存放插入的记录

头文件				
记录0	10101	Srinivasan	Comp. Sci.	65000
记录1				
记录2	15151	Mozart	Music	40000
记录3	22222	Einstein	Physics	95000
记录4				
记录5	33456	Gold	Physics	87000
记录6				
记录7	58583	Califieri	History	62000
记录8	76543	Singh	Finance	80000
记录9	76766	Crick	Biology	72000
记录10	83821	Brandt	Comp. Sci.	92000
记录11	98345	Kim	Elec. Eng.	80000

图 10-7 删除了第 1、4 和 6 条记录的图 10-4 中的文件

10101	Srinivasan	Comp. Sci.	65000	
12121	Wu	Finance	90000	
15151	Mozart	Music	40000	
22222	Einstein	Physics	95000	
32343	El Said	History	60000	
33456	Gold	Physics	87000	
45565	Katz	Comp. Sci.	75000	
58583	Califieri	History	62000	
76543	Singh	Finance	80000	
76766	Crick	Biology	72000	
83821	Brandt	Comp. Sci.	92000	
98345	Kim	Elec. Eng.	80000	
32222	Verdi	Music	48000	

图 10-11 执行插入后的顺序文件

3) 顺序文件  
存储的数据  
如何更新?

空闲列表组织被删除后的空间

删除记录

插入记录

但需定期执行重组!

4) 什么是多表聚集文件，物理存储如何实现，有何好处？

### 2. 多表聚集文件组织 p. 260

多表聚集文件组织：是指将多个关系的数据组织在一个文件中（它们的记录相互交织在一起）

*department*

<i>dept_name</i>	<i>building</i>	<i>budget</i>
Comp. Sci.	Taylor	100000
Physics	Watson	70000

*instructor*

<i>ID</i>	<i>name</i>	<i>dept_name</i>	<i>salary</i>
→ 10101	Srinivasan	Comp. Sci.	65000
→ 33456	Gold	Physics	87000
→ 45565	Katz	Comp. Sci.	75000
→ 83821	Brandt	Comp. Sci.	92000

*department* 和  
*Instructor* 的  
多表聚集文件！

Comp. Sci.	Taylor	100000
45564	Katz	75000
10101	Srinivasan	65000
83821	Brandt	92000
Physics	Watson	70000
33456	Gold	87000

注意：实现多表聚集文件组织，需底层操作系统配合，实现对文件的管理（只有大型数据库系统才支持）




5) 多表聚集文件组织方式的优点和不足?

### \*多表聚集文件组织的优缺点<sub>p. 261</sub>

- 优点:  
Good for queries involving *department* ⋈ *instructor*, and for queries involving one single department and its instructors
- 不足:  
Bad for queries involving only *department* results in variable size records
- 改进思路:  
Can add **pointer chains** to link records of a particular relation

Comp. Sci.	Taylor	100000	
455645	Katz	75000	
10101	Srinivasan	65000	
83821	Brandt	92000	
Physics	Watson	70000	
33456	Gold	87000	



### 3. 物理设计的任务

前面, 介绍了大型数据库管理系统(DBMS)支持的文件中记录的物理存储方式, 实际上都是由DBMS自动实现对用户定义好的逻辑数据(关系)的自动存储, 似乎物理存储对用户是透明的, 就像大家上机所感受的一样。

数据库应用设计人员还需要做物理设计工作吗, 如何做?

6) 实际应用中, 物理设计到底做什么?

数据库物理设计(应用开发中)的主要工作:

- 1) 在定义关系模式时, 需要确定采用定长还是变长记录;  
(通过确定采用的属性类型, 因有变长属性)
- 2) 对每个关系模式, 需要确定影响记录存放次序的搜索码;  
(根据常用/重要的查询要求, 确定主码或建聚集索引Cluster Index)
- 3) 对每个关系模式, 需要确定是否还需要建立辅助索引文件;  
(根据常用/重要的查询要求, 确定建哪些索引Index)
- 4) 对具有连接条件的一组关系模式, 需要确定是否采用多表聚集文件存储;  
(根据多表连接上重要应用查询快速访问需要)
- 5) 对应用的所有关系模式, 需要确定应当划分为多少个数据库来存储;  
(根据关系模式间相关性、应用相关性、数据保密需要、数据库备份需要等)
- 6) 对每个数据库, 需要确定数据库文件存放的物理路径(不同服务器, 不同介质)  
(根据访问效率需要、应用相关性、数据重要性等)

## 三 数据字典

讨论: RDBMS  
管理什么数据?

根据右边案例, 分析  
RDBMS会存储哪些数据?

自描述

course_id	title	dept_name	credits
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3
EE-181	Intro. to Digital Systems	Elec. Eng.	3
FIN-201	Investment Banking	Finance	3
HIS-351	World History	History	3
MU-199	Music Video Production	Music	3
PHY-101	Physical Principles	Physics	4

图 2-2 course 关系

ID	name	dept_name	salary
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

图 2-1 instructor 关系



# 三 数据字典

1)什么是元数据?

到目前为止,我们只考虑了关系本身的表示。一个关系数据库系统需要维护关于关系的数据,如关系的模式等。一般来说,这样的“关于数据的数据”称为元数据(metadata)。

关于关系的关系模式和其他元数据存储在称为数据字典(data dictionary)或系统目录(system catalog)的结构中。系统必须存储的信息类型有:

2)数字典存储什么信息?

- 关系的名字。
- 每个关系中属性的名字。
- 属性的域和长度。
- 在数据库上定义的视图的名字和这些视图的定义。
- 完整性约束(例如,码约束)。

View(虚表的原因)

此外,很多系统为系统的用户保存了下列数据:

- 授权用户的名字。
- 关于用户的授权和账户信息。
- 用于认证用户的密码或其他信息。

查询优化的基础

数据库可能还会存储关于关系的统计数据和描述数据。例如:

- 每个关系中元组的总数。
- 每个关系所使用的存储方法(例如,聚簇或非聚簇)。

反映物理存储的情况

数据字典也会记录关系的存储组织(顺序、散列或堆),和每个关系的存储位置:

3)数字典存储的作用?

- 如果关系存储在操作系统文件中,数据字典将会记录包含每个关系的文件名。
- 如果数据库把所有关系存储在一个文件中,数据字典可能将包含每个关系中记录的块记在如链表这样的数据结构中。

是RDBMS和DBA管理数据库的基础手段

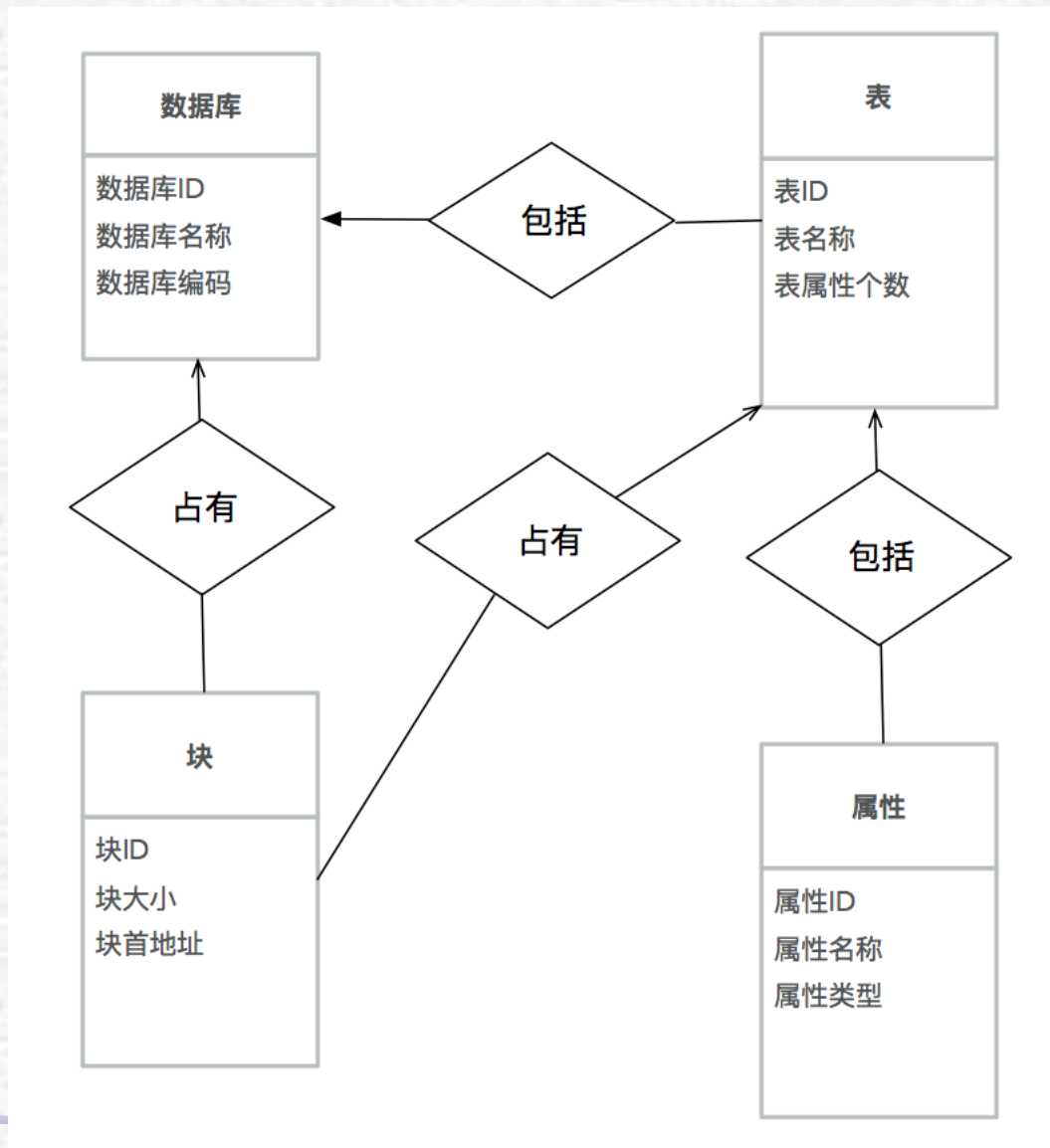
特点:关系数据库管理系统-RDBMS,采用自描述方式!  
数据存储采用关系模式,元数据存储也采用关系模式!

# 数据库存储机制

讨论: RDBMS  
元数据如何管理?

一个简单的ER图示:

一个数据库对应n个块  
一个数据库 多个表



# 四 数据库缓冲区

## 1. 数据库缓冲区及作用 P. 262

讨论：数据库缓冲区在改善数据访问中的作用？

1) 需要建立数据库缓冲区吗，为何能提高访问效率？

数据库系统的一个主要目标就是尽量减少磁盘和存储器之间传输的块数目。减少磁盘访问次数的一种方法是在主存储器中保留尽可能多的块。这样做的目标是最大化要访问的块已经在主存储器中的几率，这样就不再需要访问磁盘。

- 1) CPU处理信息快捷，但从磁盘读取记录缓慢。
- 2) 缓冲区一次I/O读多硬盘上多个记录(按块)，可明显减少磁盘I/O开销(连续读-节省时间)；  
例如：查询所有学生的记录；
- 3) 缓冲区中的记录，可能为多个应用所需要，可明显减少磁盘I/O开销(重复读-浪费时间)。  
例如：大家同时查询奥运会最新100米跑成绩

因为主存储器中保留所有的块是不可能的，所以需要管理主存储器中用于存储块的可用空间的分配。缓冲区(buffer)是主存储器中用于存储磁盘块的拷贝的那一部分。每个块总有一个拷贝存放在磁盘上，但是在磁盘上的拷贝可能比在缓冲区中的拷贝旧。负责缓冲区空间分配的子系统称为缓冲区管理器(buffer manager)。

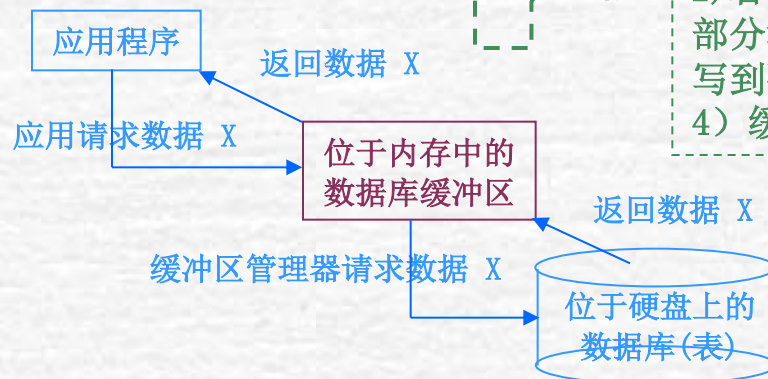


## 2. 数据库缓冲区管理器 P. 262

- 1) 若X不在缓冲区中，则为其分配空间，并向硬盘(硬盘上物理数据库)请求 X；
- 2) 若缓冲区已满，则按某种测量策略清除部分块，清除前先将块中最新修改数据回写到硬盘(数据库)！
- 4) 缓存区从硬盘(数据库)读取 X；

2) 如何管理数据库缓冲区？

图表描述：



文字描述：

当数据库系统中的程序需要磁盘上的块时，它向缓冲区管理器发出请求(即调用)。如果这个块已经在缓冲区中，缓冲区管理器将这个块在主存储器中的地址传给请求者。如果这个块不在缓冲区中，缓冲区管理器首先在缓冲区中为这个块分配空间，如果需要的话，会把其他块移出主存储器，为这个新块腾出空间。移出的块仅当它自从最近一次写回磁盘后被修改过才被写回磁盘。然后缓冲区管理器把请求的块从磁盘读入缓冲区，并将这个块在主存储器中的地址传给请求者。缓冲区管理器的内部动作对发出磁盘块请求的程序是透明的。

# 缓冲区管理器

- 比虚拟存储器管理更复杂的技术支持
  - 缓冲区替换策略(最近最少使用, LRU)
  - 不允许写回磁盘的块 (如故障发生时, 正更新操作的块)
  - 强制写出块 (故障发生时, 强制写出缓冲区的块, 以防丢失数据)

### 3. 缓冲区替换策略P. 263

Select \* from instructor natural join department

- 外表instructor（立即丢弃策略）
  - 每个记录处理完就结束
  - 处理完一个完整的块，就可以换出
- 内表department（最近最常用MRU策略）
  - 对外表里的每一个记录，需要检查内表的每一个块。
  - 每当一个内表块处理完，要到其他所有内表块处理完才再次访问
  - 最近最常使用的块将是最后一个要再次访问的块，可以先换出
  - 使用中的块不能换出
- 索引块一般不换出
- 大多数数据库系统都采用LRU策略

```

for each instructor 中的元组 i do
  for each department 中的元组 d do
    if i[dept_name] = d[dept_name]
      then begin
        令 x 为下列式子定义的元组:
        x[ID] := i[ID]
        x[dept_name] := i[dept_name]
        x[name] := i[name]
        x[building] := d[building]
        x[budget] := d[budget]
        将元组 x 包含进 instructor ⋈ department 的结果中
      end
    end
  end
end

```

图 10-17 计算连接的过程



## 3. 缓冲区替换策略 P. 263

3) 如何确定缓冲区中的无用区块?

对缓冲区中的块替换策略而言，目标是减少对磁盘的访问。对通用程序来说，精确预言哪个块将被访问是不可能的。因此，操作系统使用过去的块访问模式来预言未来的访问。通常我们假定最近访问过的块最有可能再一次被访问。因此，如果必须替换一个块，则替换最近访问最少的块。这种方法称为最近最少使用 (Least Recently Used, LRU) 块替换策略。

数据库：假设该例子中的两个关系存储在不同的文件中。在这个例子中，我们可以看到，一旦 *instructor* 中的一个元组处理过，这个元组就不再需要了。因此一旦处理完 *instructor* 元组构成的一个完整的块，这个块就不再需要存储在主存储器中了，尽管它刚刚使用过。一旦 *instructor* 块中最后一个元组处理完毕，就应该命令缓冲区管理器释放这个块所占用的空间。这个缓冲区管理策略称为立即丢弃 (loss-immediate) 策略。

数据库：现在考虑包含 *department* 元组的块。对 *instructor* 关系中的每个元组，我们需要检查 *department* 元组的每个块。当一个 *department* 块处理完毕后，我们知道这个块要到所有其他 *department* 块处理完才会被再次访问。因此，最近最常使用的 *department* 块将是最后一个要再次访问的块，最近最少使用的 *department* 块是接着要访问的块。这个假设正好与构成 LRU 策略基础的假设相反。实际上，上述过程中块替换的最优策略是最近最常使用 (Most Recently Used, MRU) 策略。如果必须从缓冲区中移除一个 *department* 块，MRU 策略将选择最近最常使用的块(当块被使用时不能被替换)。



# 随堂小测试

- 在文件中，记录有哪两种方式表示，对于记录的集合如何组织？
- 什么是元数据？元数据如何存储？

# 课后小结和作业安排

- 基本知识：
  - 定长记录与变长记录
  - 文件中记录的组织方式
  - 数据字典，元数据存储机制
  - 数据库缓冲区
- 延展性学习：
  - 元数据如何管理
- 作业

第10章习题：10.4，10.15，10.18 。



# 下一讲的学习内容

## 学习任务：各种索引技术

- 数据库中索引的作用
- 主要索引技术（顺序索引、B+树索引、散列）
- 索引技术之间的区别