上海交通大学
SHANGHAI JIAO TONG UNIVERSITY

SJTU

数 据 库 技 术

郭 捷

(guojie@sjtu.edu.cn)

饮 水 思 源 • 爱 国 荣 校

第九章 数据库存储管理

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

目 录

数据组织

- 以最优的形式在外存上组织、存放庞大数据集。
  - 最优：
    - ✓存储效率高，节省存储空间；
    - ✓存取效率高，访问速度快，代价小；

01

数据库的逻辑组织方式与物理组织方式

SJTU

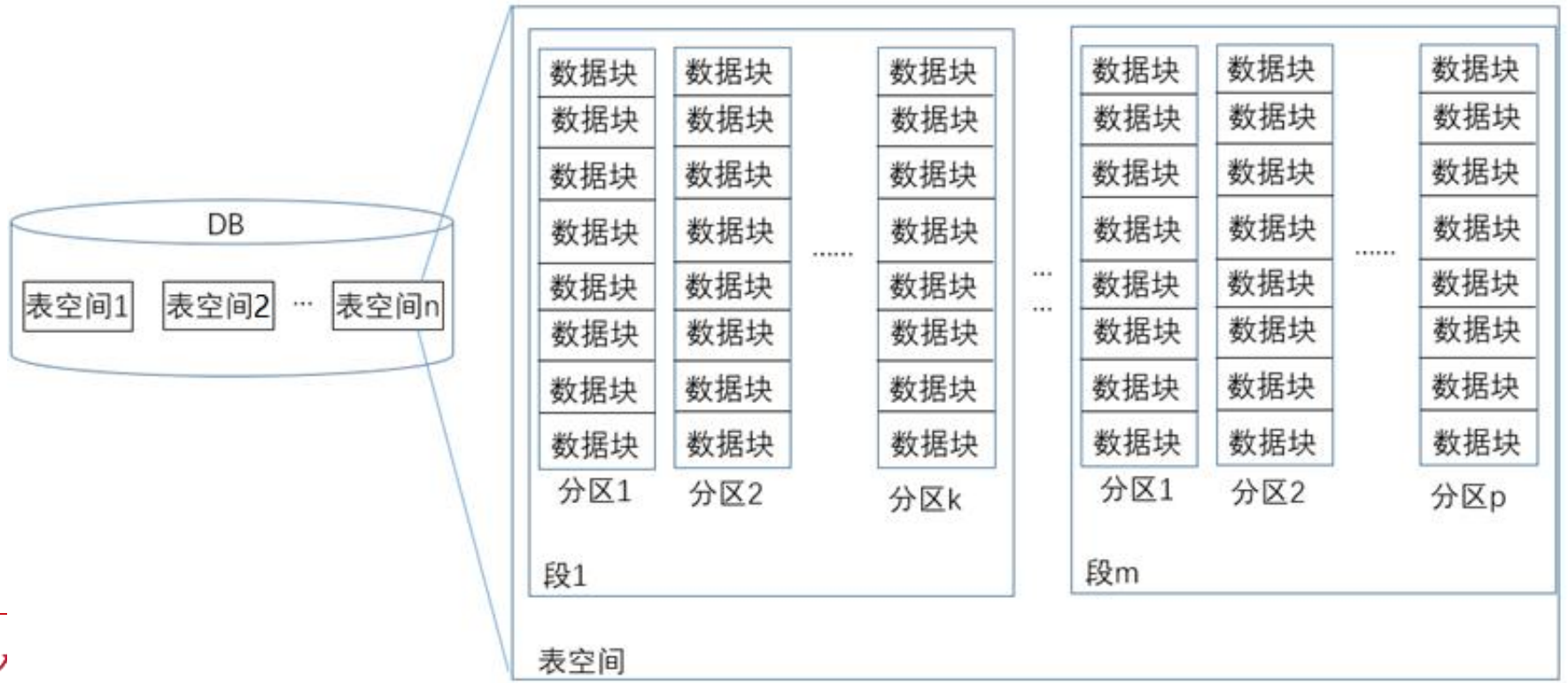
数据存储的管理方式

❖数据存储的管理方式

- ■方式一：每个DB对象（基本表、索引）对应一个操作系统文件；
  - ●e.g. PostgreSQL, Kingbase ES
- ■方式二：整个DB对应一个或若干个文件（段页式存储）
  - ●e.g. Oracle, SQL Server

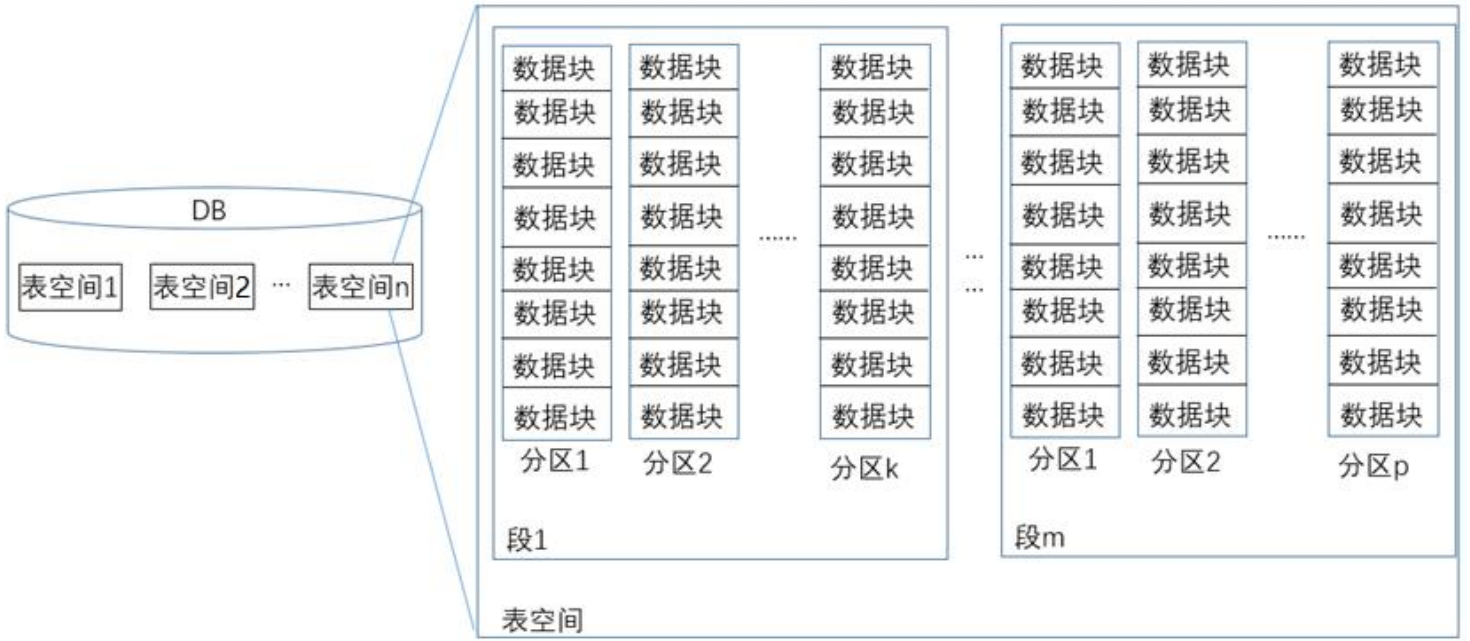
数据库的逻辑组织

- 数据库的逻辑组织方式（方便数据管理）：
  - 表空间----段----分区----数据块



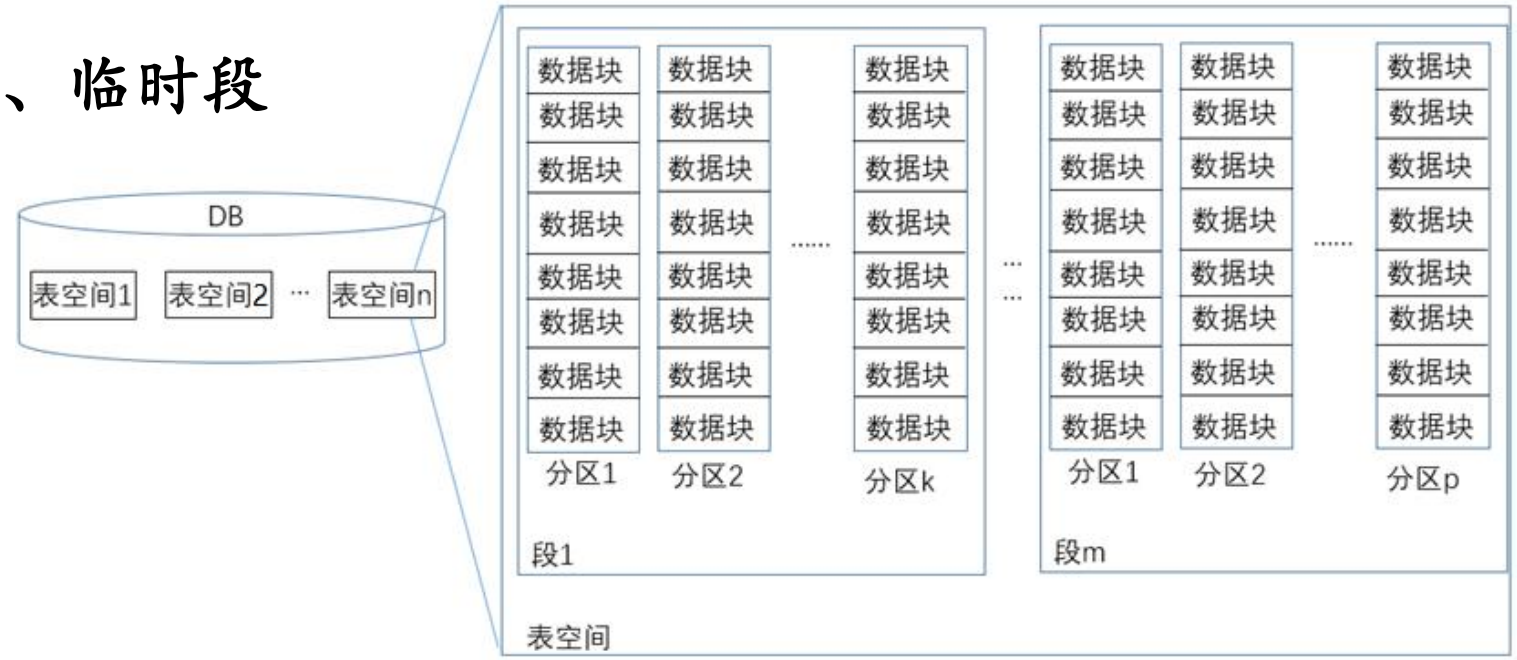
数据库的逻辑组织

- 表空间（Tablespace）：
  - 磁盘上的一个或多个物理文件，<u>一个物理文件只能属于一个表空间</u>。
  - 一个数据库可以有多个表空间，从逻辑上组织数据库的数据存储；
    - ✓ e.g. 系统表空间、联机表空间、临时表空间



数据库的逻辑组织

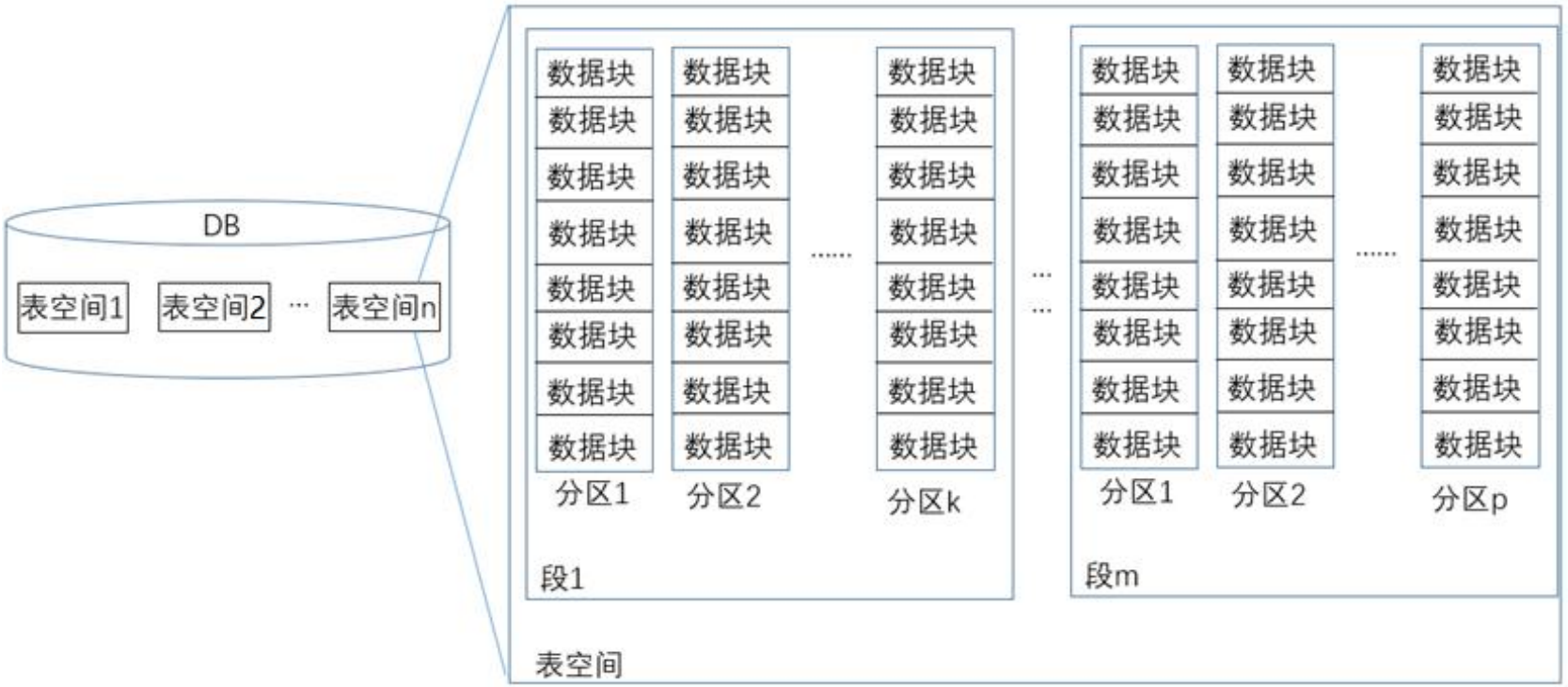
- 数据段（Segment）：
  - 一个表空间逻辑上由多个段组成，每个段可以逻辑上组织不同类型的数据。
    - ✓ 例如：数据段、索引段、临时段



数据库的逻辑组织

分区（Extent）：

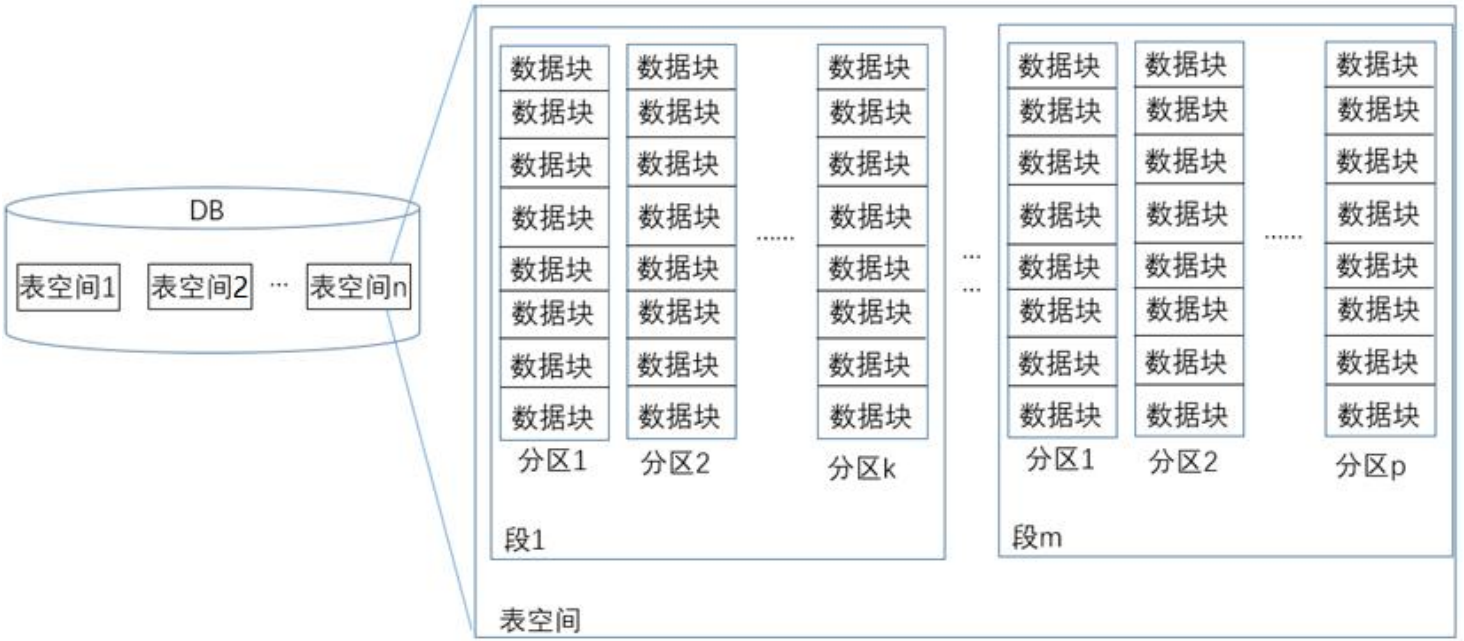
- 一个段逻辑上由多个分区组成，每个分区由一组连续数据块组成。



数据库的逻辑组织

数据块（Block）：

- 数据库的磁盘存取单元，大小为操作系统块的整数倍。



数据库的物理组织

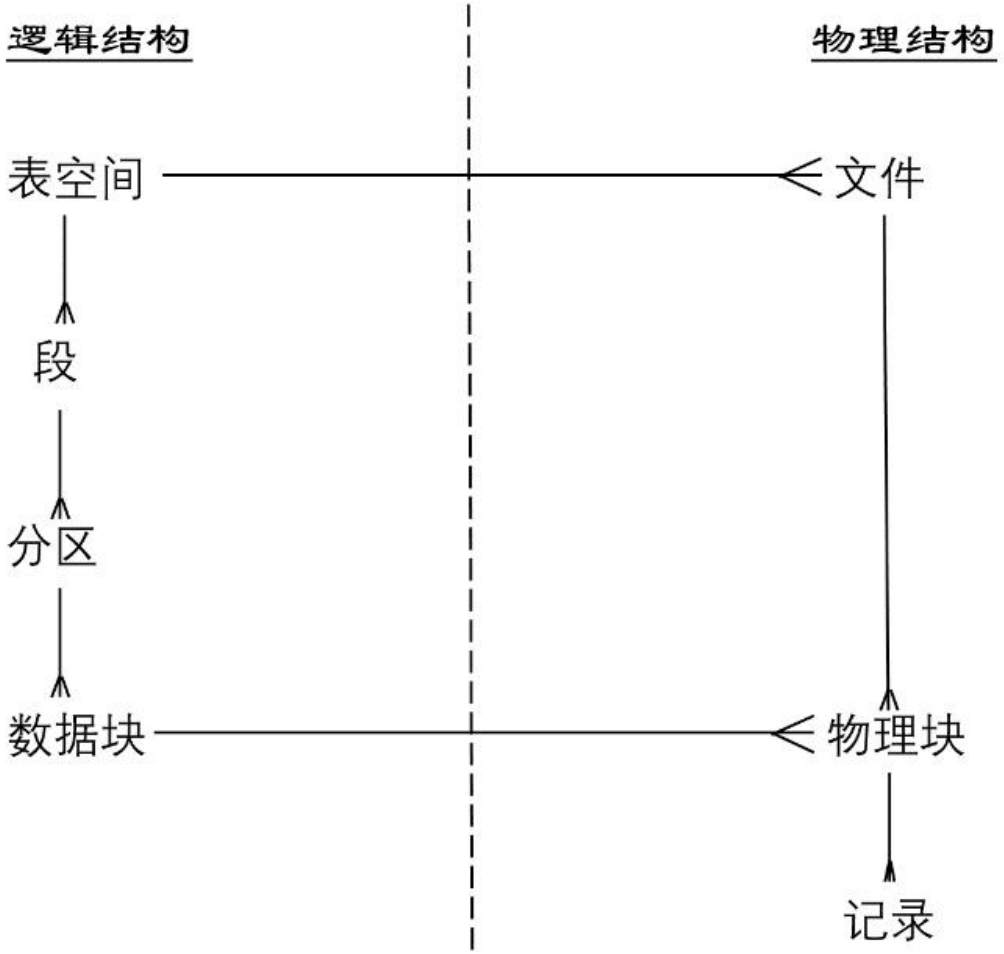
- 数据库的物理组织方式：
  - 文件-----块-----记录

- 文件：
  - 操作系统文件
- 记录：
  - 物理块中存放的多条元组（记录）

- 块：
  - 每个文件物理上分成定长的存储单元，即操作系统的物理块。
  - 存储分配和I/O的基本单位

逻辑组织与物理组织的对应关系

- 数据库逻辑组织方式与物理组织方式的对应关系：



02

记录表示

记录表示

- 元组存储：
  - 形式一：定长记录存储
  - 形式二：变长记录存储

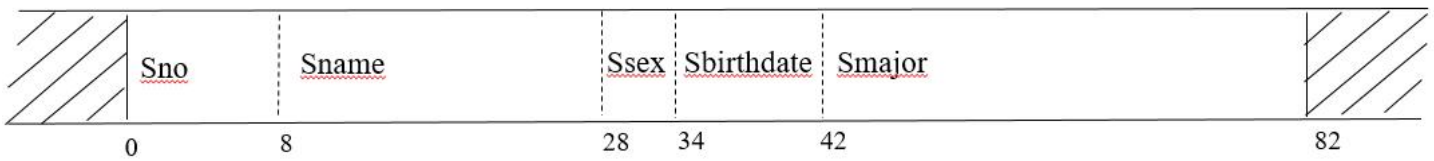
一、定长记录存储

- 定长记录存储：
  - 关系表中的每条记录占据相同大小的空间；
  - 变长字段以定长形式存储，预留最大长度空间；

■ 例：

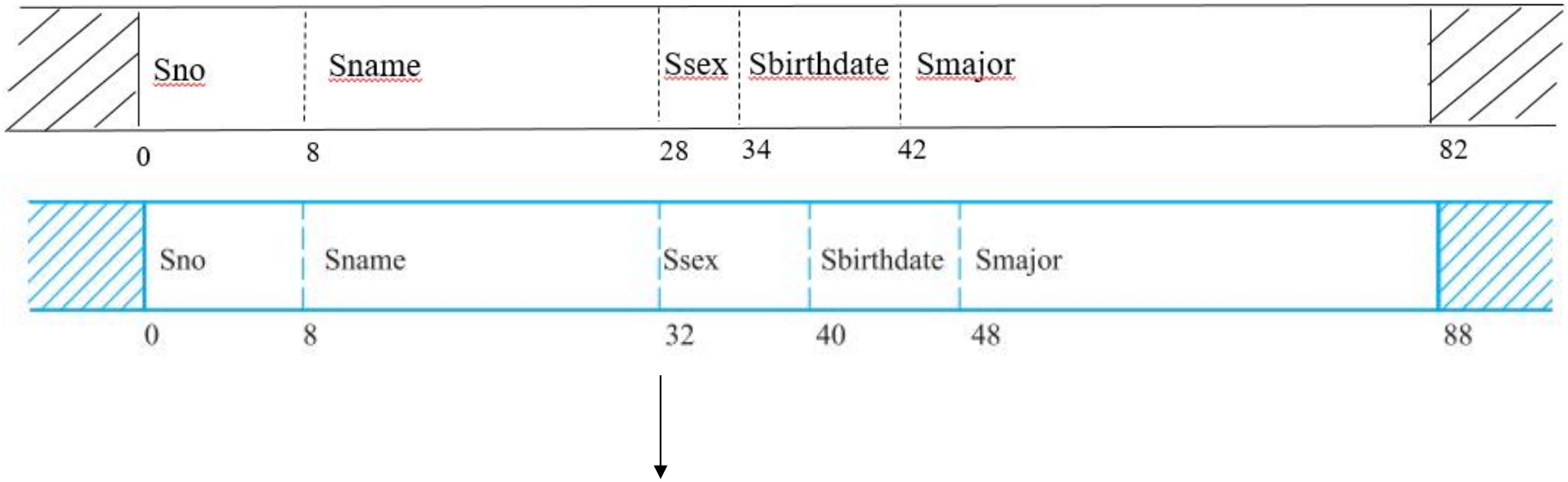
```
CREATE  TABLE  Student (
    Sno  CHAR(8) PRIMARY KEY,
    Sname VARCHAR(20) UNIQUE,
    Ssex  CHAR(6),
    Sbirthdate  Date,
    Smajor  VARCHAR(40) );
```

VARHCHAR类型属性
分配最大长度空间



一、定长记录存储

- 有些硬件系统对内存数据的起始地址有要求：4或8的倍数。
  - 在外存中保证各字段的起始地址是4或8的倍数



1. 每个字段都是从8的倍数的地址起始
2. Sname是20字节，但分配了24字节
3. Ssex定义了6字节，但分配了8字节

一、定长记录存储

- 定长存储的优劣：
  - 优势
    - ✓ 快速定位到记录及其属性的物理位置；
    - ✓ 增删改比较方便快捷；
  - 劣势
    - ✓ 浪费存储空间；

二、变长记录存储

- 变长记录：
  - 存储要求
    - ✓能快速访问一条记录
    - ✓能快速访问记录中所有属性（包括定长和变长属性）；
  - 变长记录存放有三种方式。

二、变长记录存储

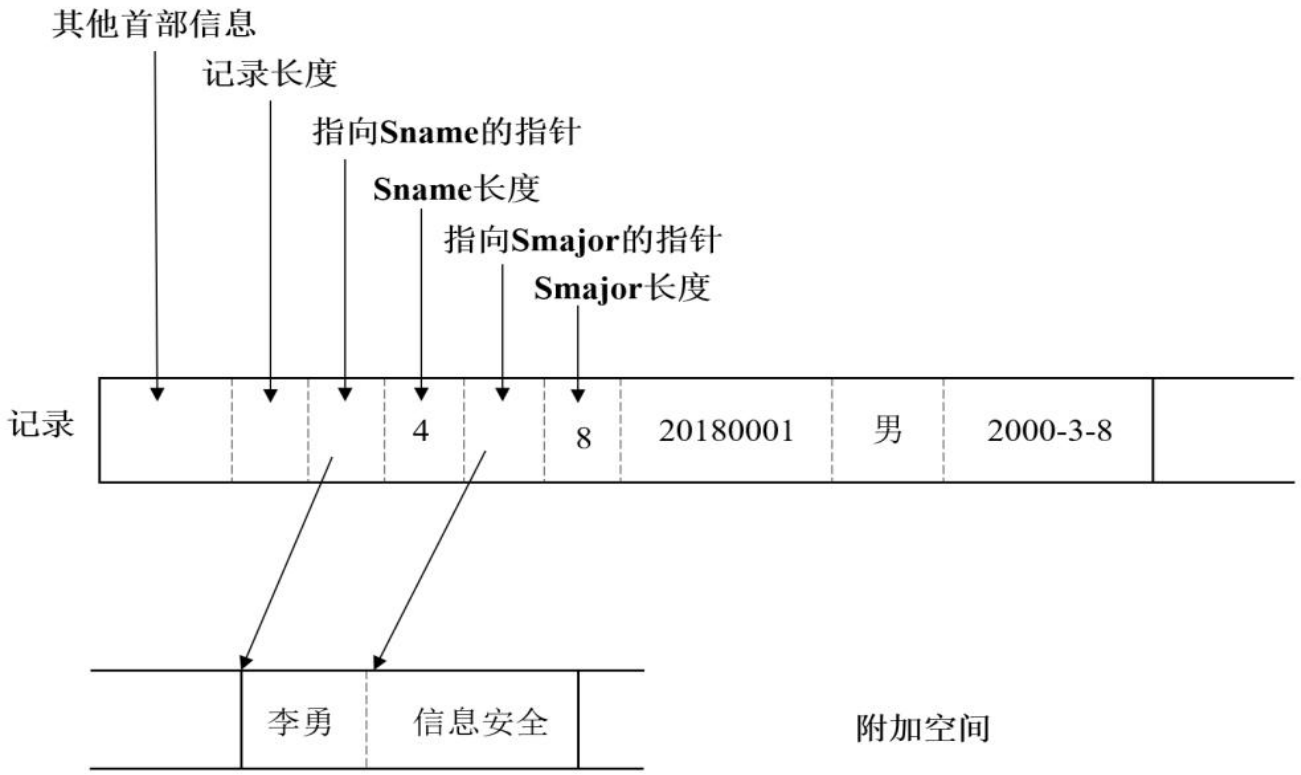
- 方式一：
  - 在每条记录的头部记录该条记录的长度。
  - 在记录的每个变长字段前记录该字段的长度。

二、变长记录存储

- 方式二：
  - 先存放定长字段，再存放变长字段
  - 第一个变长字段紧随定长字段，从第二个变长字段开始，在记录首部用指针（偏移量）指向变长字段。

二、变长记录存储

- 方式三：
  - 将定长字段与变长字段分开存储在不同的块中；
  - 多用于BLOB（Binary Large Object）等类型数据存储；
  - 若经常访问定长字段，可减少数据存取时的I/O数量。



03

块的组织

块的组织

- 定长记录存储的块组织
- 变长记录存储的块组织

一、定长记录存储的块组织

定长记录存储的块组织：

- 表中元组依次存放在块中；
- 首部+记录+空闲空间

- 首部：块头信息、块ID、最后一次修改和访问该块的时间戳、每条记录在块内的偏移量、空闲空间头指针；

一、定长记录存储的块组织

- 定长记录存储的块维护：
  - 增：在空闲空间直接插入新元组；
  - 改：直接在原位置修改；
  - 删：回收空间，将空闲空间加入空闲空间链表，不需要移动；

一、定长记录存储的块组织

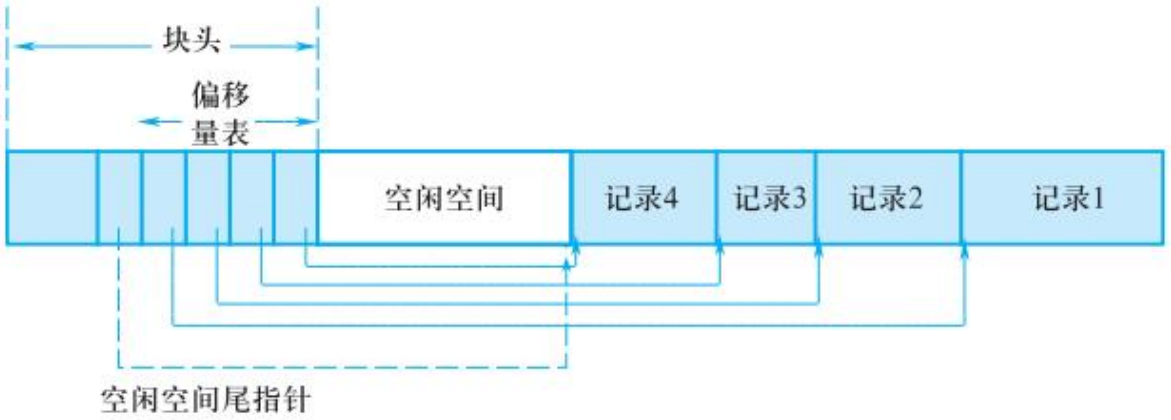
- 在Student表中删除2018003和2018006两条元组。

二、变长记录存储的块组织

- 变长记录存储的块组织：

  - 表中元组从块的尾部连续存放；

  - 首部+空闲空间+记录



    - 首部：各记录的指针（块内偏移量）、空闲空间尾指针（块内偏移量）；

二、变长记录存储的块组织

- 变长记录存储块的维护：
  - 增
    - 从空闲空间尾部分配空间；
    - 在偏移量表中记录改元组的起始位置；
    - 调整空闲空间尾指针；

二、变长记录存储的块组织

变长记录存储块的维护：

■ 删

- ●在偏移量表中为该元组指针置删除标记
- ●释放元组空间，移动物理位置在其前面的元组（保证空闲空间连续）
- ●修改指针
  - ✓被移动元组在偏移量表中的指针
  - ✓空闲空间尾指针；

二、变长记录存储的块组织

- 变长记录存储块的维护：
  - 改
    - 在原位置修改；
    - 若修改后记录在原位置放不下，会带来记录的迁移；

04

关系表的组织

关系表的组织

- 关系表的5种存放方式：
  - 堆存储
  - 顺序存储
  - 多表聚簇存储
  - B+树存储
  - 哈希存储

一、堆存储

- 表中的一条记录可以存放在该表的任何块中，没有顺序要求；
- 插入元组时，在该表的块中找到合适的空闲空间即可；
- 如果没有足够的空闲空间，就为该表申请新的块；

二、顺序存储

- 一个表中的各条记录根据指定的属性或属性组的取值大小顺序的存放；
- 同一个表的不同块之间通过指针链接实现有序；

二、顺序存储

顺序存储的优劣：

- 优势：可以高效地处理按排序属性（组）进行查询的请求
  - 例：如果Student表按专业Smajor的升序存放，则可以快速回答如下SQL语句：

```
SELECT Sno, Sname, Ssex
FROM Student
WHERE Smajor='计算机科学与技术'
```

- 劣势：维护代价较高（在增改时需要移动已有的记录以保序）

三、多表聚簇存储

- 不同表的元组聚簇存放在同一组块中，减少连接操作：
  - 例：两个具有主外码的参照关系表Student和SC，按照学号Sno相等聚簇存放

三、多表聚簇存储

- 多表聚簇存储的优劣：
  - 优势：可以减少连接操作带来的开销，快速回答如下查询

```
SELECT SC.Sno, Sname, count(*)
FROM Student, SC
WHERE Student.Sno=SC.sno AND SC.Sno='20180001';
```

三、多表聚簇存储

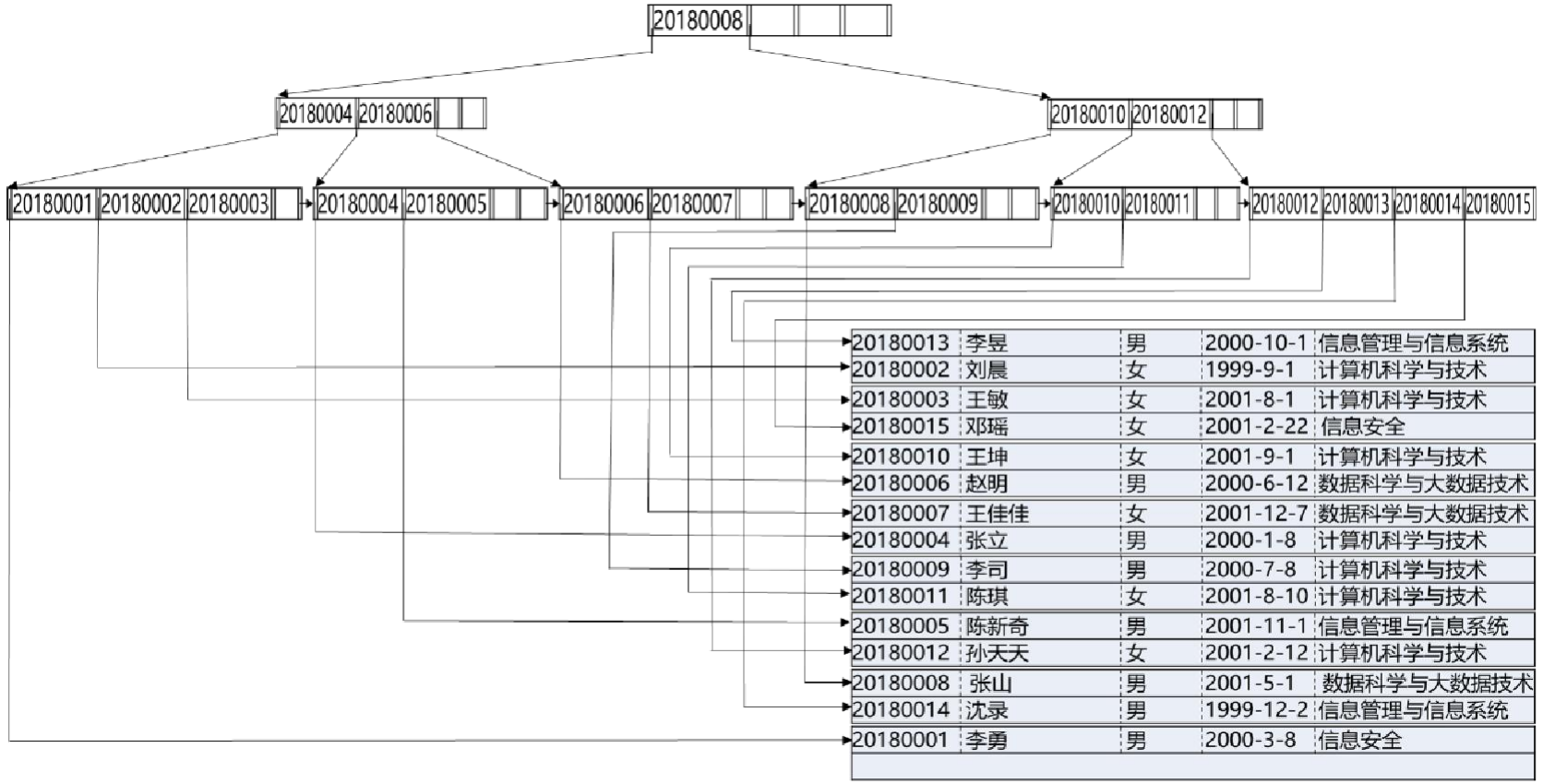
- 多表聚簇存储的优劣：
  - 劣势：
    - 降低某些查询的查询效率（同一表中的元组会分散在更多块中）

```
SELECT * FROM Student
WHERE Smajor=’数据科学与大数据技术’ AND Ssex=’女’;
```

    - 在更新操作时会带来更频繁的数据迁移

四、B+树存储

- 以B+树索引的方式确定记录存放在哪个数据块中；
- 保持较高的访问效率的同时，降低数据维护开销：



五、哈希存储

- 用哈希函数计算表中指定属性的哈希值，以此确定相应记录放在哪个块中。
  - 哈希表：由B个哈希桶（bucket）组成，每个桶编号0到B-1，对应一个或多个物理块，存放一条或多条记录；
  - 哈希函数：输入为记录的哈希属性，输出为介于0到B-1之间的整数，对应哈希桶号；
- 将记录的存储位置与其指定属性的哈希值之间建立一个对应关系，查找记录时可快速定位这条记录。

第九章 数据库存储管理

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

目 录

索引结构

- 索引——类似图书目录的附加结构，提高查询处理效率；

  - 优势：

    - ✓表的索引块数量通常比数据块数量少得多；
    - ✓可以用高效的方法快速查找索引块；
    - ✓若索引文件足够小，可长期驻留内存缓冲区，加少I/O操作；

索引结构

- 索引——类似图书目录的附加结构，提高查询处理效率；
  - 劣势：索引会带来额外的开销
    - ✓ 存储索引开销
    - ✓ 建立索引开销
    - ✓ 维护索引开销；

01

顺序表索引

顺序表索引

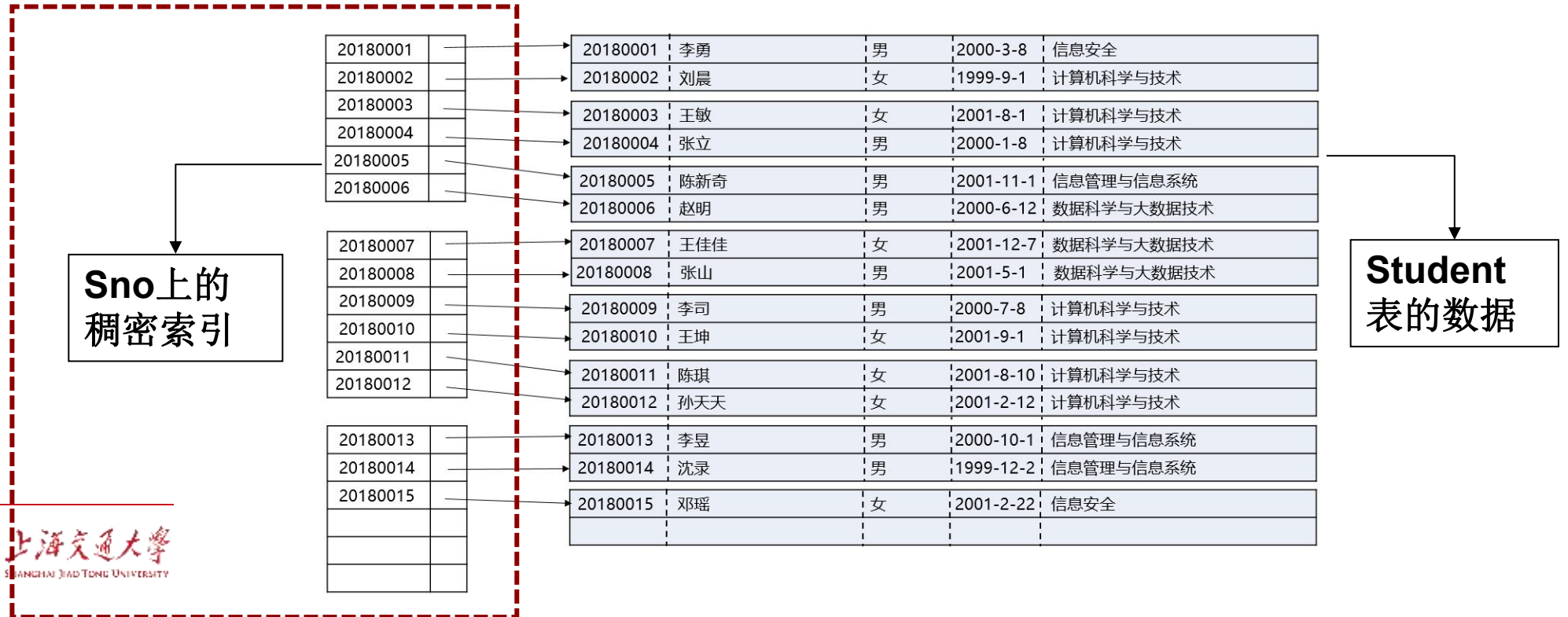
- 在顺序表的排序属性（组）上建立索引；
- 也称作主索引（Primary Index）或聚簇索引（Clustering Index）；
  - 稠密索引
  - 稀疏索引
  - 多级索引

一、稠密索引

稠密索引（Dense Index）

- 索引块中存放每条记录的索引属性值以及指向相应记录的指针；

例：假设每个块存放2条记录或6个Sno索引项



一、稠密索引

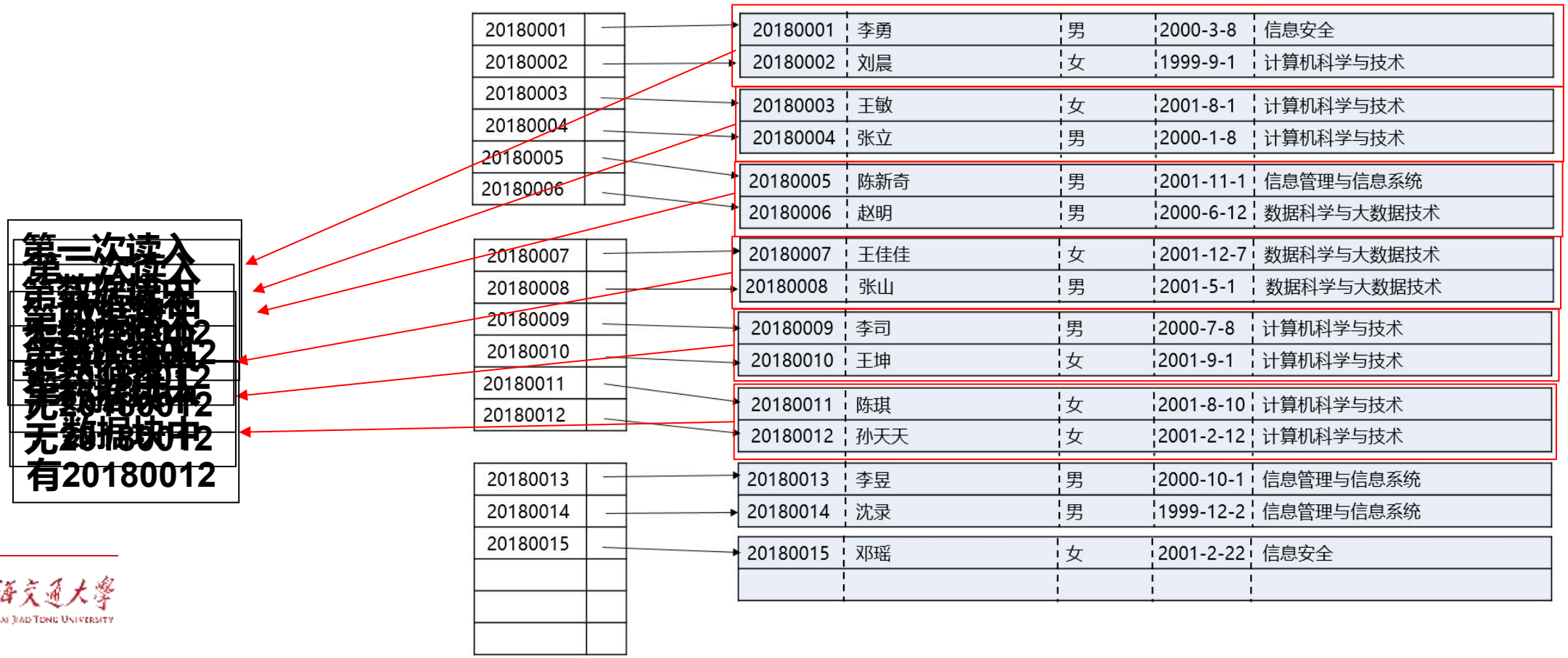
- 有序索引，索引较大时，可利用二分查找法在稠密索引查找指定索引项；
- 利用索引查询可减少I/O；

例：如果用户想在Student表中查找学号为20180012的学生信息

```
Seletct * Form Student
WHERE Sno=’ 20180012’;
```

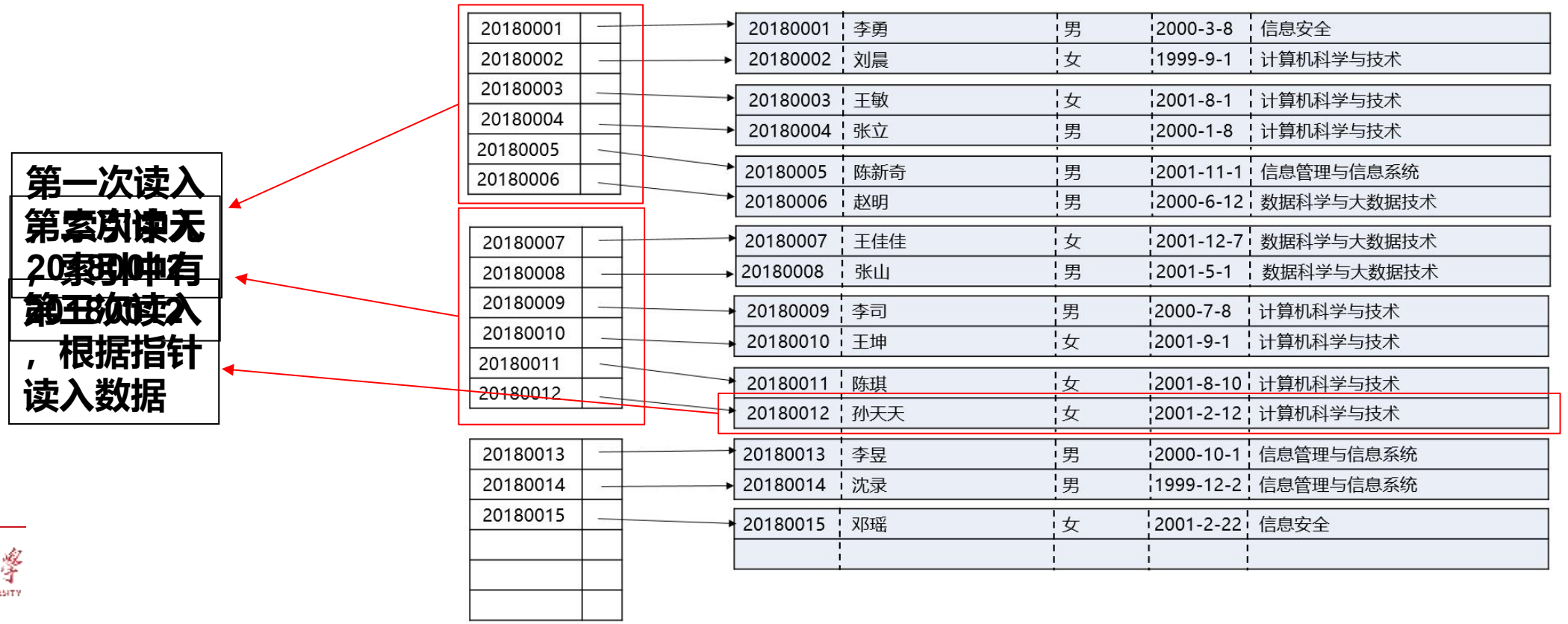
一、稠密索引

- 直接查找，不借助索引，6次I/O
  - 依次读入6个数据块，6次



一、稠密索引

- 利用稠密索引，3次I/O
  - 读入索引块（顺序查找索引），2次
  - 根据指针读取20180012元组，1次

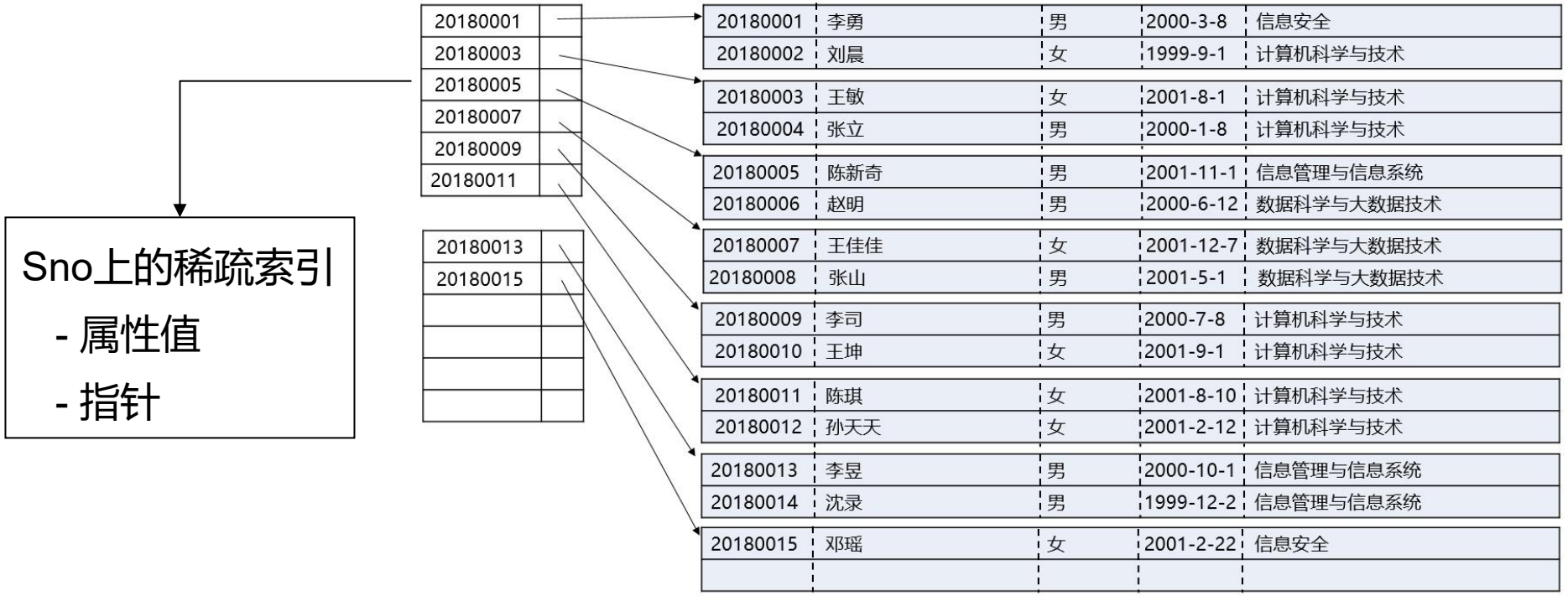


二、稀疏索引

稀疏索引（Sparse Index）

- 基本表的每个物理存储块只对应一个索引项；
- 每个索引项存放每个物理块的第一条记录的索引属性值及指向该物理块的指针。

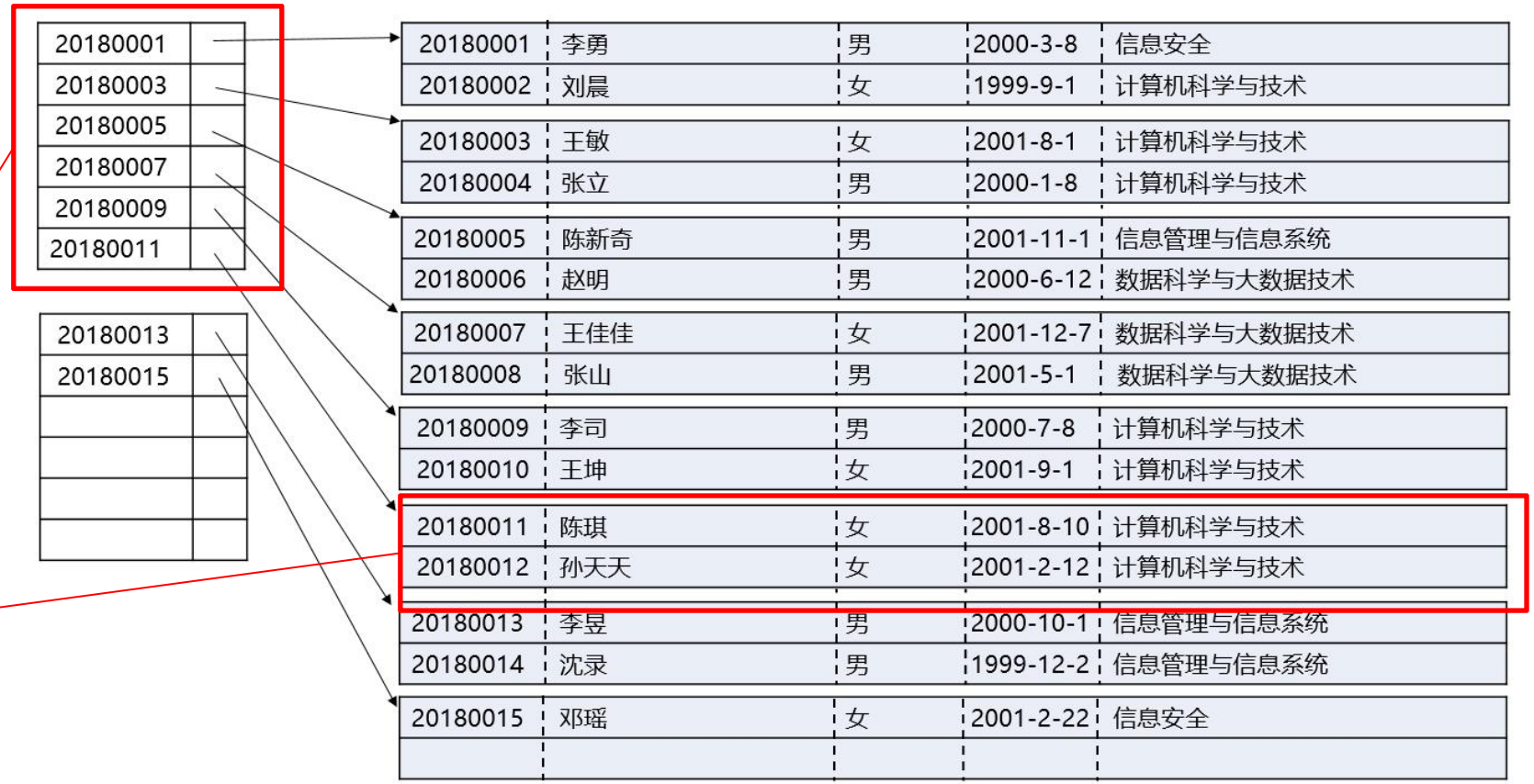
例：假设每个块存放2条记录或6个Sno索引项



二、稀疏索引

- 在Student表中查找学号为20180012的学生信息，需要2次I/O；
  - ✓读入索引块，查找属性值小于等于20180012的最大索引项（顺序或二分）
  - ✓读入数据块，找到20180012对应的元组；

```
Seletct * Form Student
WHERE Sno=' 20180012';
```



第一次读入，找到20180011

第二次读入，找到20180012

二、稀疏索引

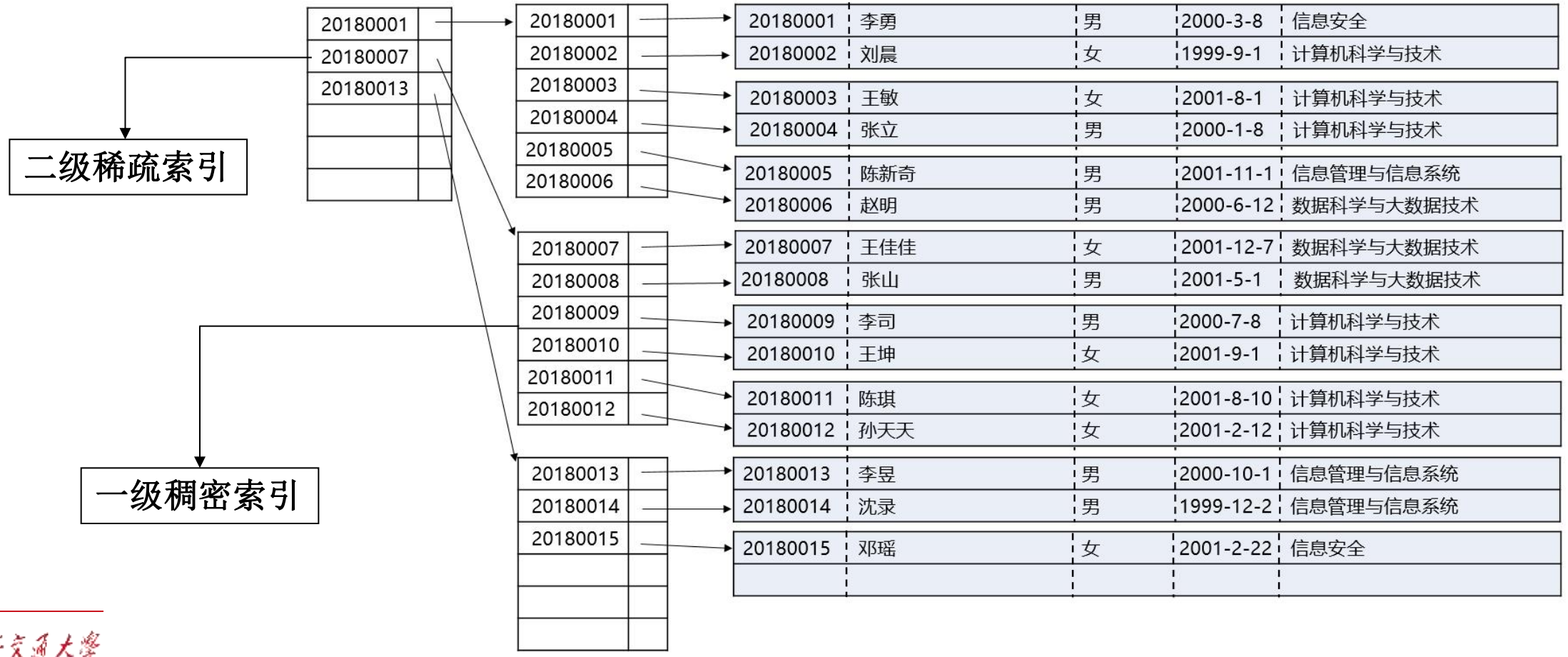
- 稀疏索引的优势：
  - 尺寸小于稠密索引
  - 索引维护代价较小
    - 对基本表进行增删改时，只要不是存储块第一条记录的索引属性，稀疏索引就不需要维护。

三、多级索引

- 解决索引尺寸大问题
- 索引建立：
  - 第一级索引是稠密或稀疏索引
  - 第二级及以上为建立在上一级索引上的稀疏索引
  - 重复直到尺寸合适
- 利用多级索引查找：
  - 从高级索引逐层向下，知道定位到记录所在的物理块

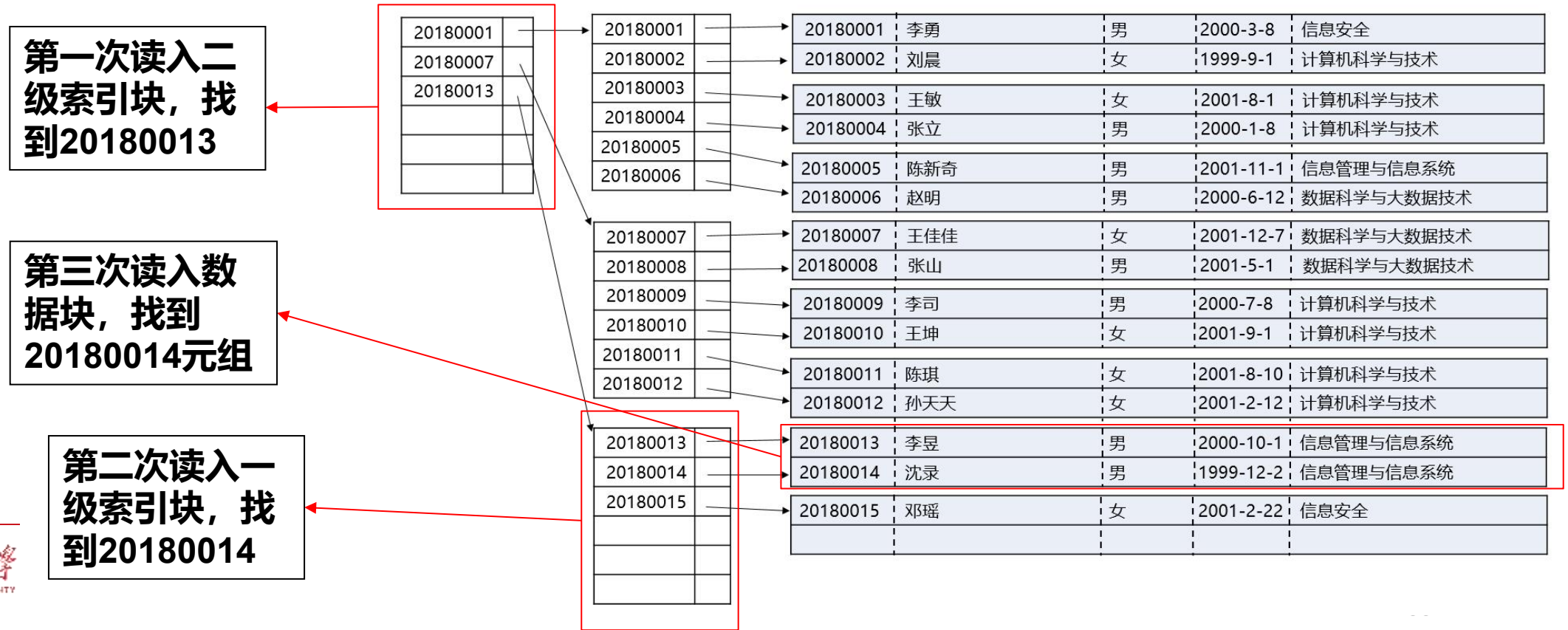
三、多级索引

多级索引（Multilevel Index）



三、多级索引

- 在Student表中查找学号为20180014的学生信息，3次I/O ；
  - ✓第一次读入二级索引块，1次I/O
  - ✓第二次读入一级索引块，1次I/O
  - ✓第三次读入数据块，1次I/O



02

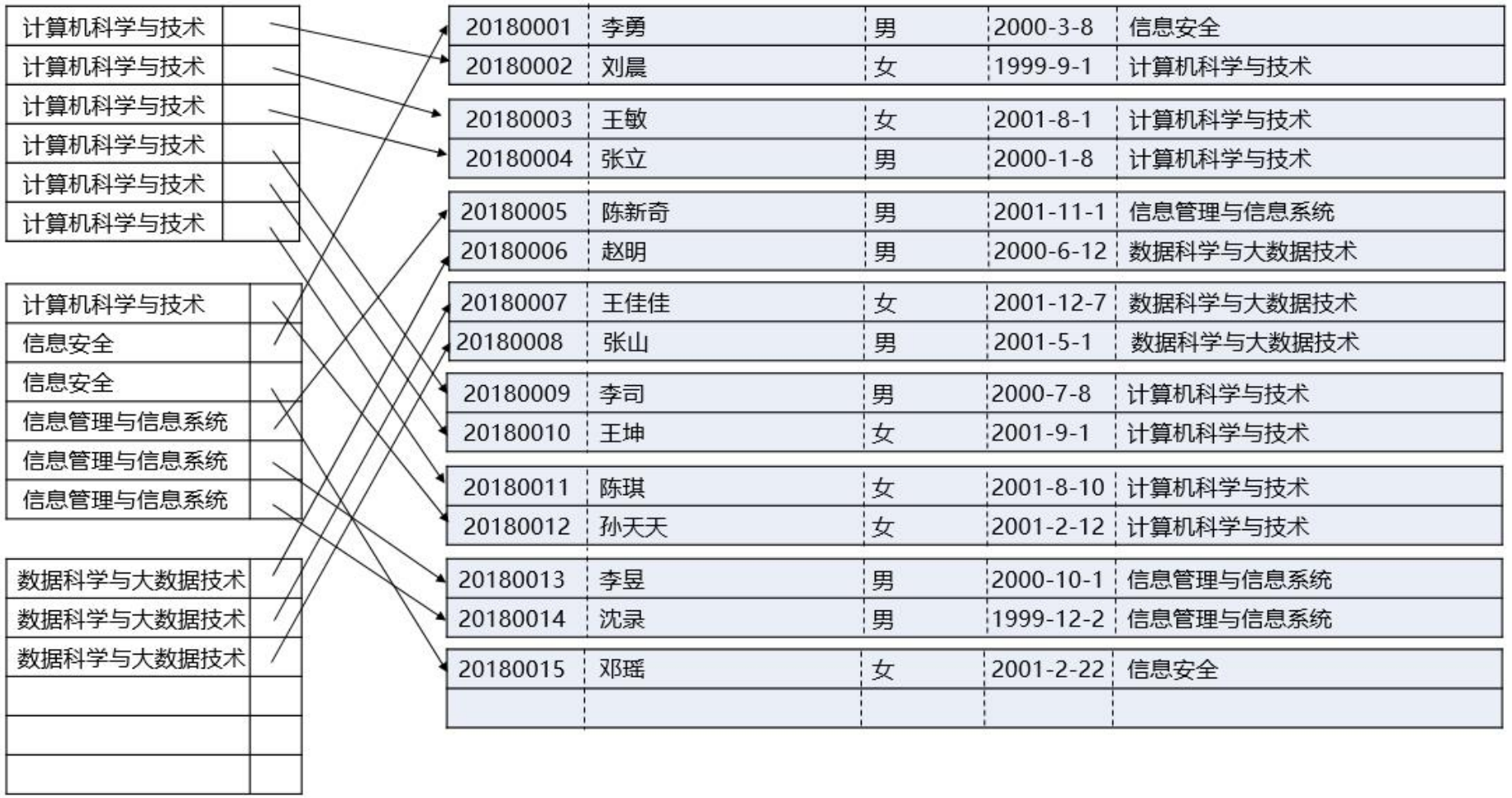
辅助索引

辅助索引

- 辅助索引（Secondary Index）
  - 建立在表的非排序属性上的索引；
  - 一个表最多只能建立一个主索引，但可以在不同属性上建立多个辅助索引；
  - 辅助索引必须是稠密索引；

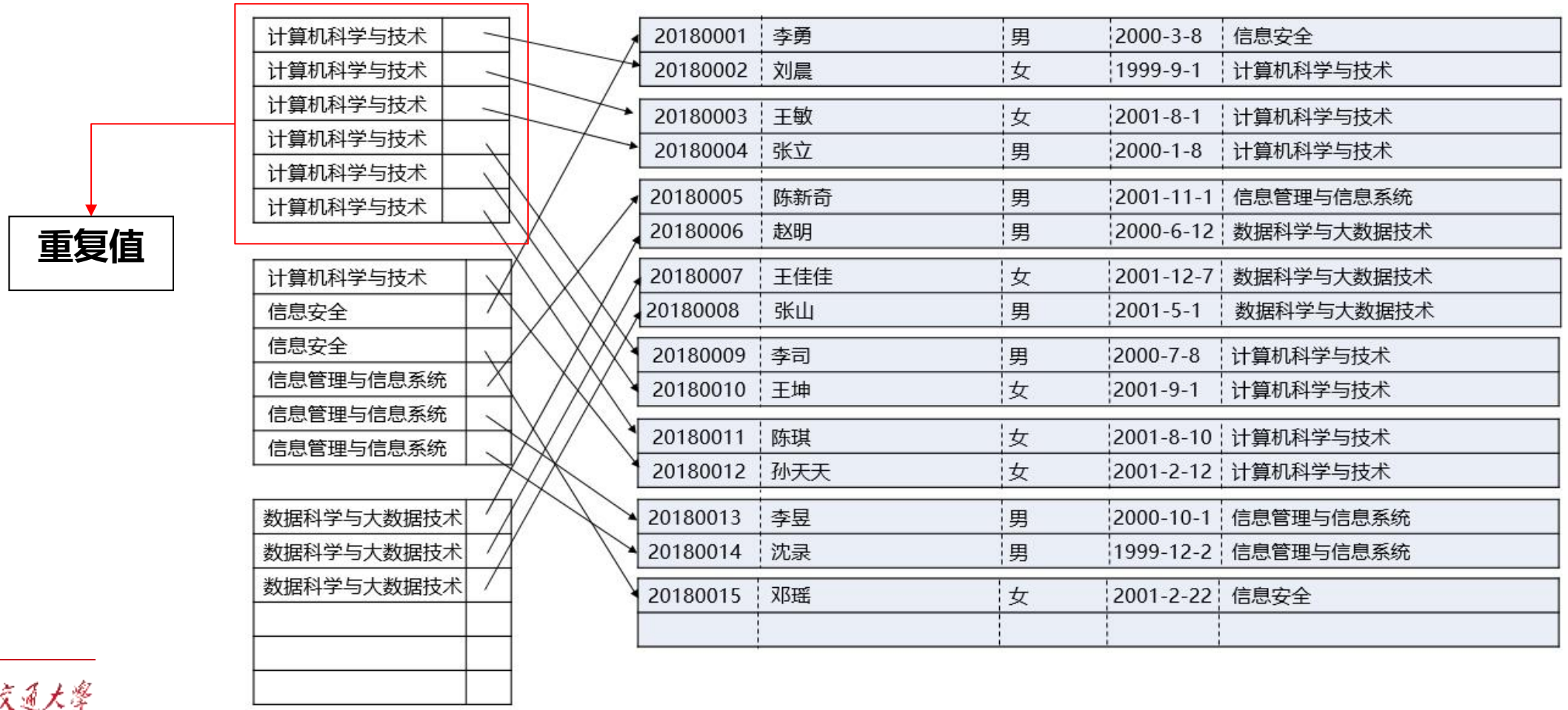
辅助索引

辅助索引（Secondary Index）



辅助索引

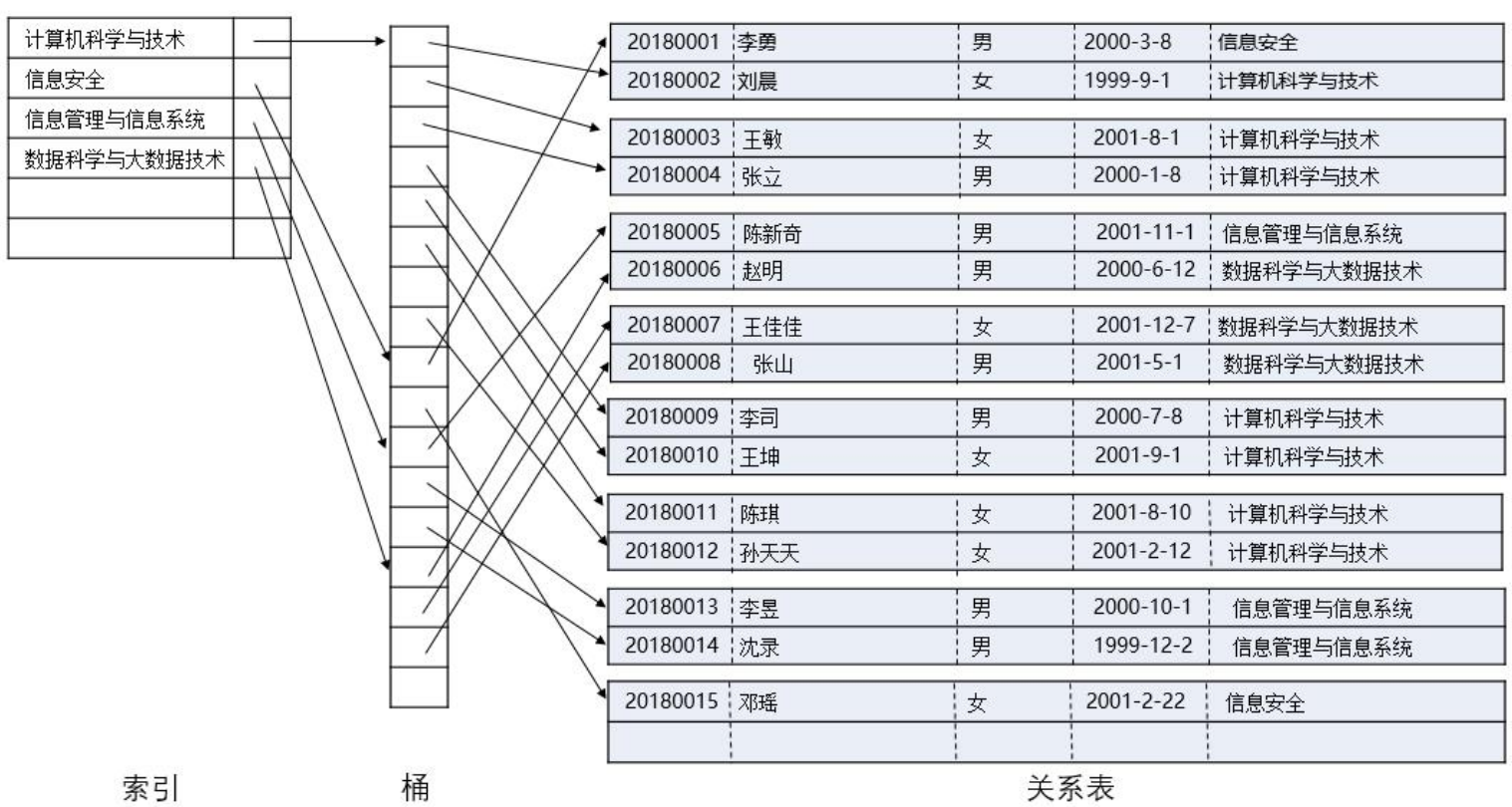
- 辅助索引的属性往往会取重复值，去掉可减小索引大小与查找开销）



辅助索引

- 引入指针桶去除重复索引项
  - 索引项指针——>指针桶相应位置——>相应的元组

例：假设一个物理块放 6个索引项，16个指针桶，2条元组



辅助索引

辅助索引（Secondary Index）

- 当在关系表上建立了多个辅助索引时，可以利用指针桶回答涉及多个属性的查询；

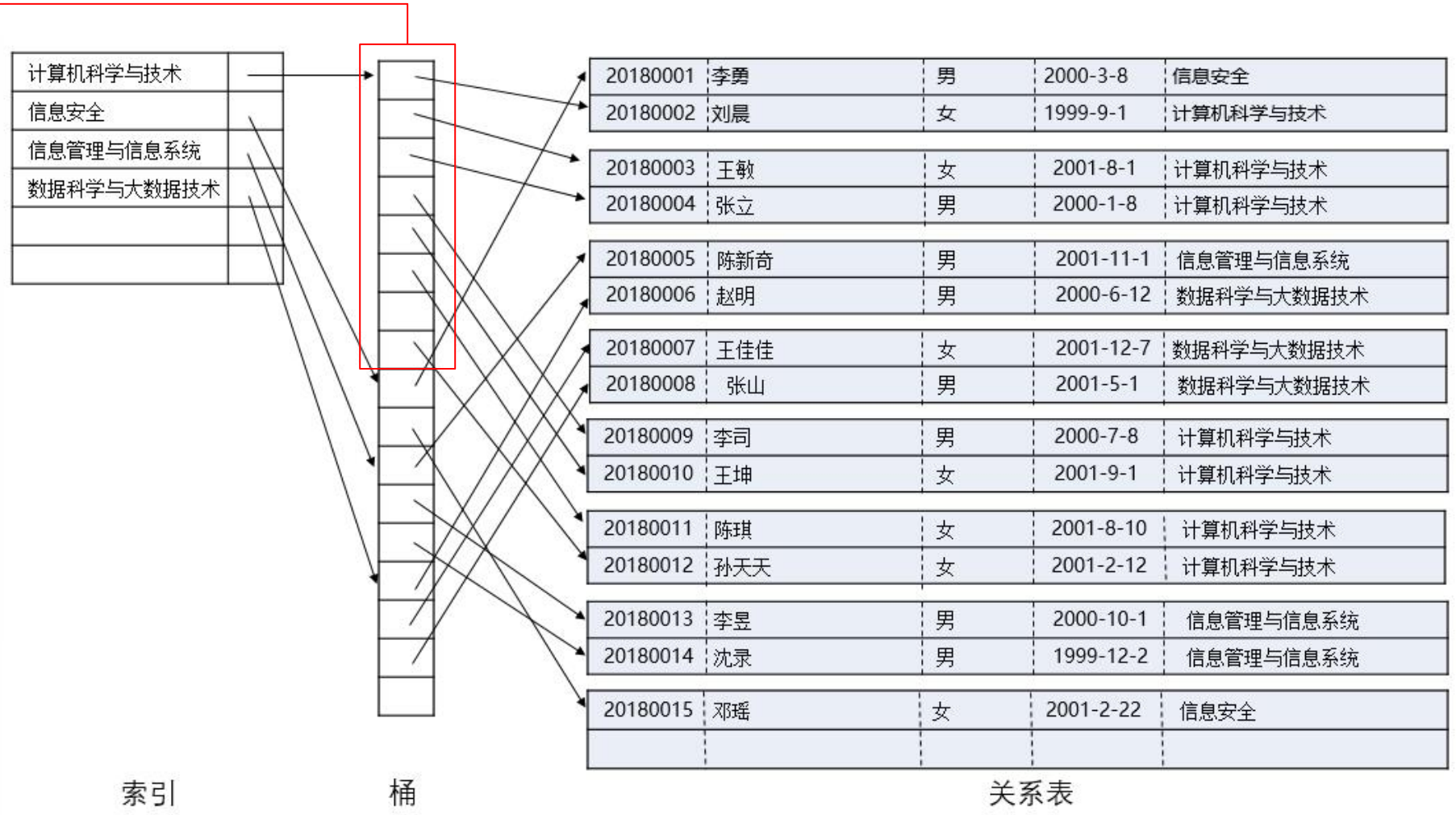
例：

```
SELECT * FROM Student
WHERE Smajor='计算机科学与技术'
AND Sbirthdate='2002-1-1';
```

- 找出满足Smajor=‘计算机科学与技术’的指针组
- 找出满足 Sbirthdate=‘2002-1-1’ 的指针组
- 两个指针组求交集

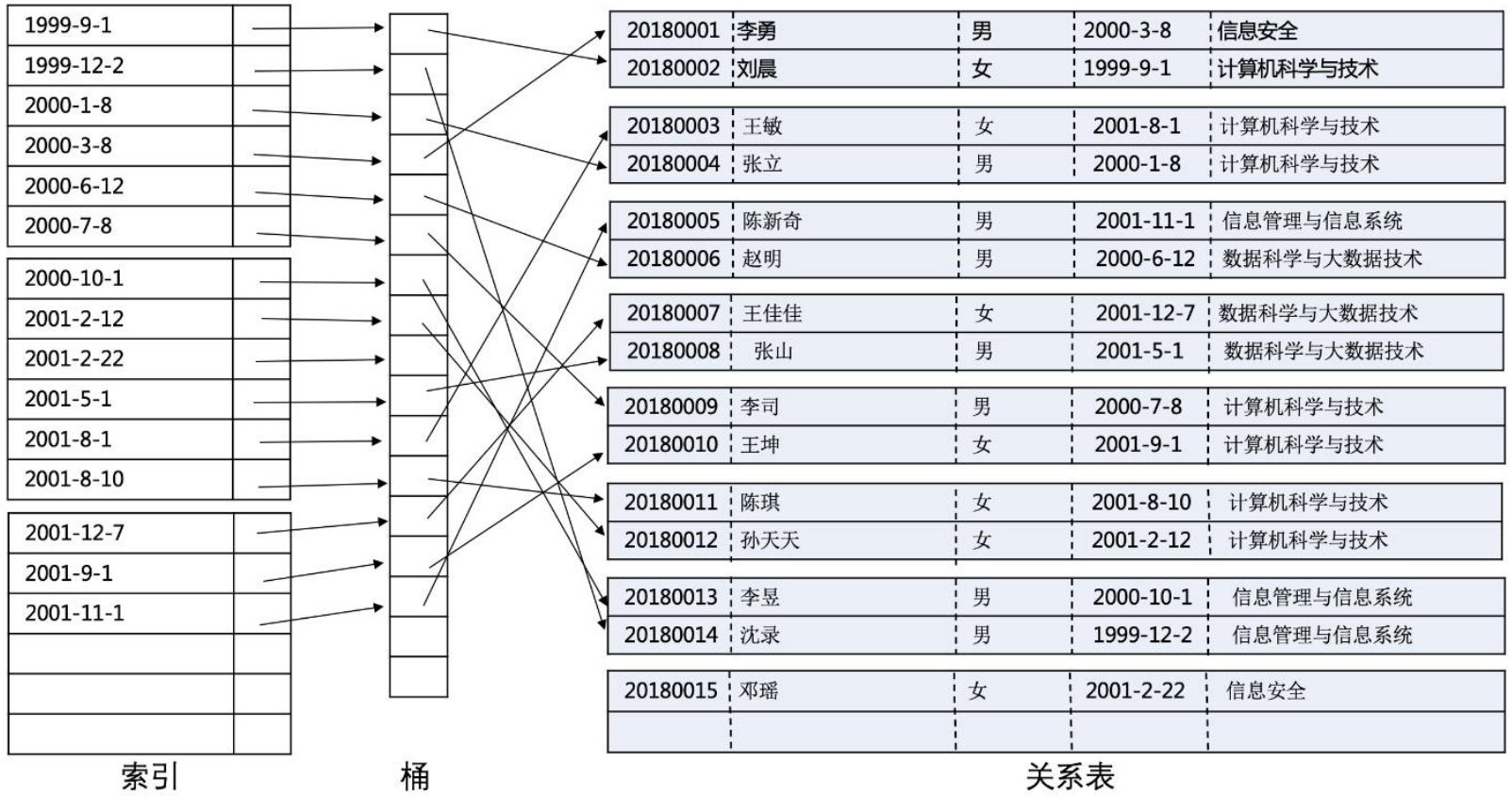
辅助索引

‘计算机科学与技术’属性值对应指针桶



辅助索引

- 2002-1-1指针桶为空
  - 两个属性值指针桶交集为空，查询结果为空；



03

B+树索引

SJTU

B+树索引

- 稠密索引与稀疏索引
  - 随着数据量增大
    - ✓索引本身庞大，查找效率不能令人满意
    - ✓按同一属性的不同值查找，时间效率可能相差大
    - ✓为保持索引项和元组有序，维护代价高
- 多级索引
  - 缓解前两个问题
  - 加剧索引维护代价

B+树索引

- B+树索引
  - 解决大型索引的组织和维护
  - 查找效率高、按不同值查找性能平衡、易于维护等

一、B+树索引的结构

- B+树索引的结构
  - 本质上是一个多级索引
  - 将索引块组织成一棵平衡树
    - 从树根到树叶的所有路径一样长
- B+树的三类结点
  - 根结点：只有一个
  - 中间结点
  - 叶结点

根结点与中间结点统称为非叶结点

一、B+树索引的结构

- B+树的秩（order）
  - 一个索引块最多能存放的指针的个数
- 一棵秩为n的B+树索引
  - 每个结点最多包含n-1个属性值key；
  - 除了根结点外，每个结点最少包含⌈(n-1)/2⌉个属性值key（根结点最少含有一项）；
  - 含有j-1项的非叶结点，有j个指针，分别指向其j个孩子（叶结点除外，它没有孩子）；
  - 所有的叶结点都在同一级上。含有j-1项的叶结点，有j个指针，前j-1个指针指向相应的关系表元组，第j个指针指向兄弟叶结点；

一、B+树索引的结构

例：B+树索引的典型结点

- 包含了n-1个属性值K_1，K_2，…，K_{n-1}
- n个指针P_1，P_2，…，P_n
- 结点中的属性值按序存放，如果i<j，则K_i < K_j ；

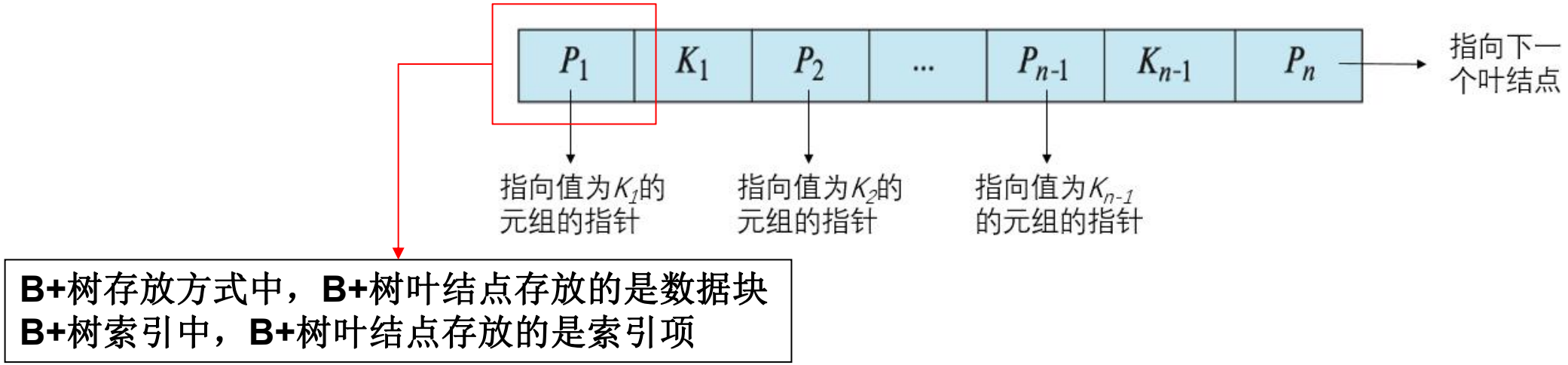
P_1	K_1	P_2	…	P_{n-1}	K_{n-1}	P_n

一、B+树索引的结构

◆ B+树索引的叶结点

✓指针P_i指向关系表中属性值为K_i的元组（i=1，2，…，n-1）

✓指针P_n指向其兄弟叶结点。最后一个叶结点的P_n为空。

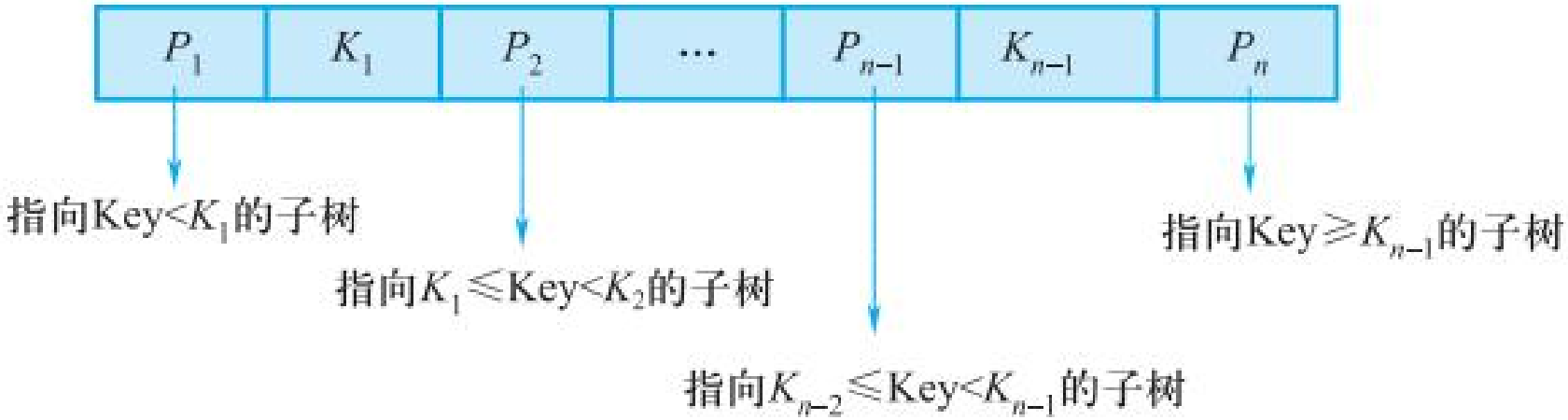


B+树存放方式中，B+树叶结点存放的是数据块
B+树索引中，B+树叶结点存放的是索引项

一、B+树索引的结构

◆ B+树索引的非叶结点

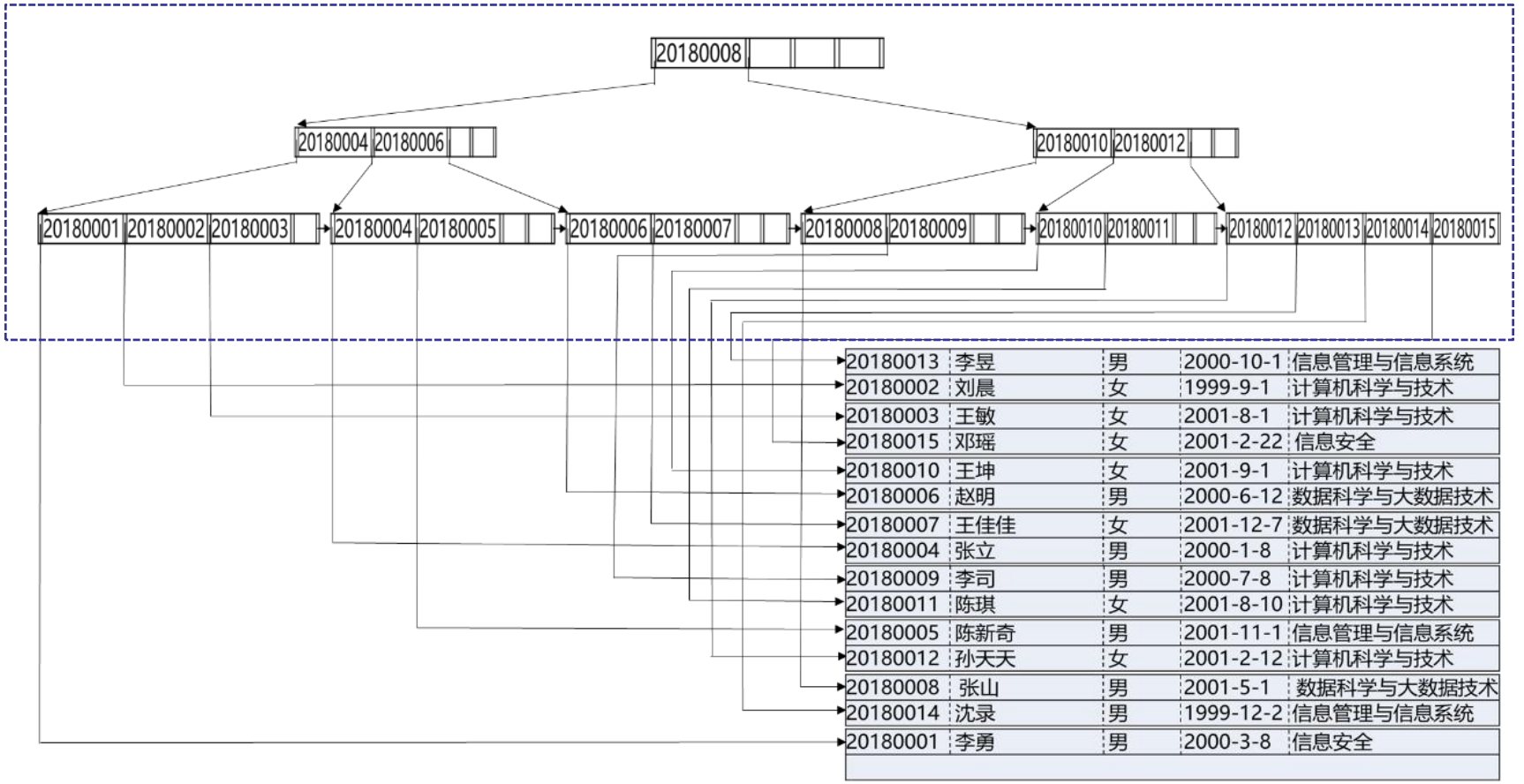
- P_i指向其下层的孩子结点（i=1，2，…，n）。
  - ✓P_1指向的子树，其所有属性值Key均满足 Key ＜ K_1
  - ✓P_i（i=2，3，…，n-1）指向的子树，其所有属性值Key均满足K_{i-1}≤ Key ＜ K_i
  - ✓P_n指向的子树，其所有属性值Key均满足Key ≥ K_{n-1}。



一、B+树索引的结构

◆ 例：Student表Sno属性上的B+树索引（n=5）

- 一个物理块：
  - 最多存放4个Sno属性值和5个指针
  - 最少存放2个Sno属性值和3个指针
  - 根结点除外
- 叶子结点形成一个顺序集合



二、B+树索引的查询

- 高效地完成随机查找和范围查询
  - 随机查找：按照索引属性的取值进行查找
    - 从根结点开始
    - 沿父子结点指针逐层向下搜索
    - 直到叶结点（匹配则返回结果，否则没有满足条件的元组）

二、B+树索引的查询

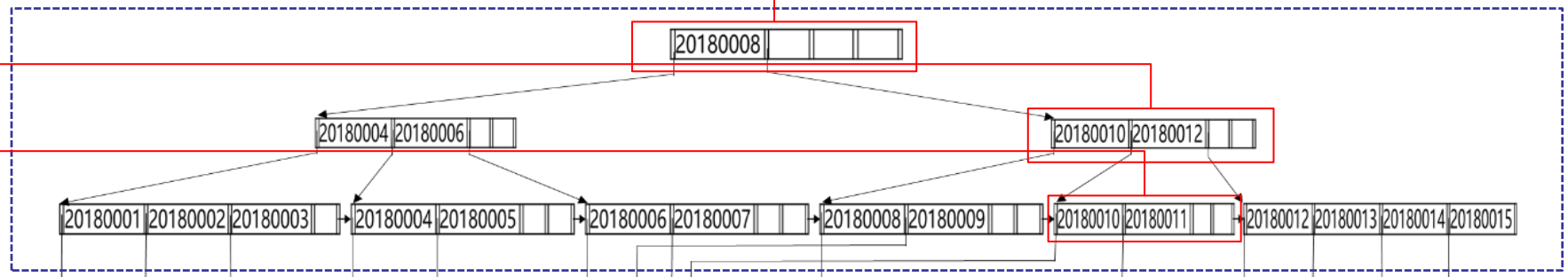
- 随机查找：按照索引属性的取值进行查找

例：在Student表中查询Sno=20180011的学生信息

20180011>20180008
沿右子树继续搜索

20180010<20180011<20180012
沿20180012左指针继续搜索

在叶结点中找到20180011
用其左侧的记录指针取出记录



20180013	李昱	男	2000-10-1	信息管理与信息系统
20180002	刘晨	女	1999-9-1	计算机科学与技术
20180003	王敏	女	2001-8-1	计算机科学与技术
20180015	邓瑶	女	2001-2-22	信息安全
20180010	王坤	女	2001-9-1	计算机科学与技术
20180006	赵明	男	2000-6-12	数据科学与大数据技术
20180007	王佳佳	女	2001-12-7	数据科学与大数据技术
20180004	张立	男	2000-1-8	计算机科学与技术
20180009	李司	男	2000-7-8	计算机科学与技术
20180011	陈琪	女	2001-8-10	计算机科学与技术
20180005	陈新奇	男	2001-11-1	信息管理与信息系统
20180012	孙天天	女	2001-2-12	计算机科学与技术
20180008	张山	男	2001-5-1	数据科学与大数据技术
20180014	沈录	男	1999-12-2	信息管理与信息系统
20180001	李勇	男	2000-3-8	信息安全

二、B+树索引的查询

- 高效地完成随机查找和范围查询

  - 范围查找（between … and …）

    - 用随机查找的方法分别找到范围条件的入口点和结束点

    - 对入口点和结束点之间的属性值进行顺序搜索

二、B+树索引的查询

- 无论查询条件中的属性值是什么，查询效率都相似

  - 例：查询Sno=20180011的学生与查询Sno=20180001的学生，其I/O次数是一样的

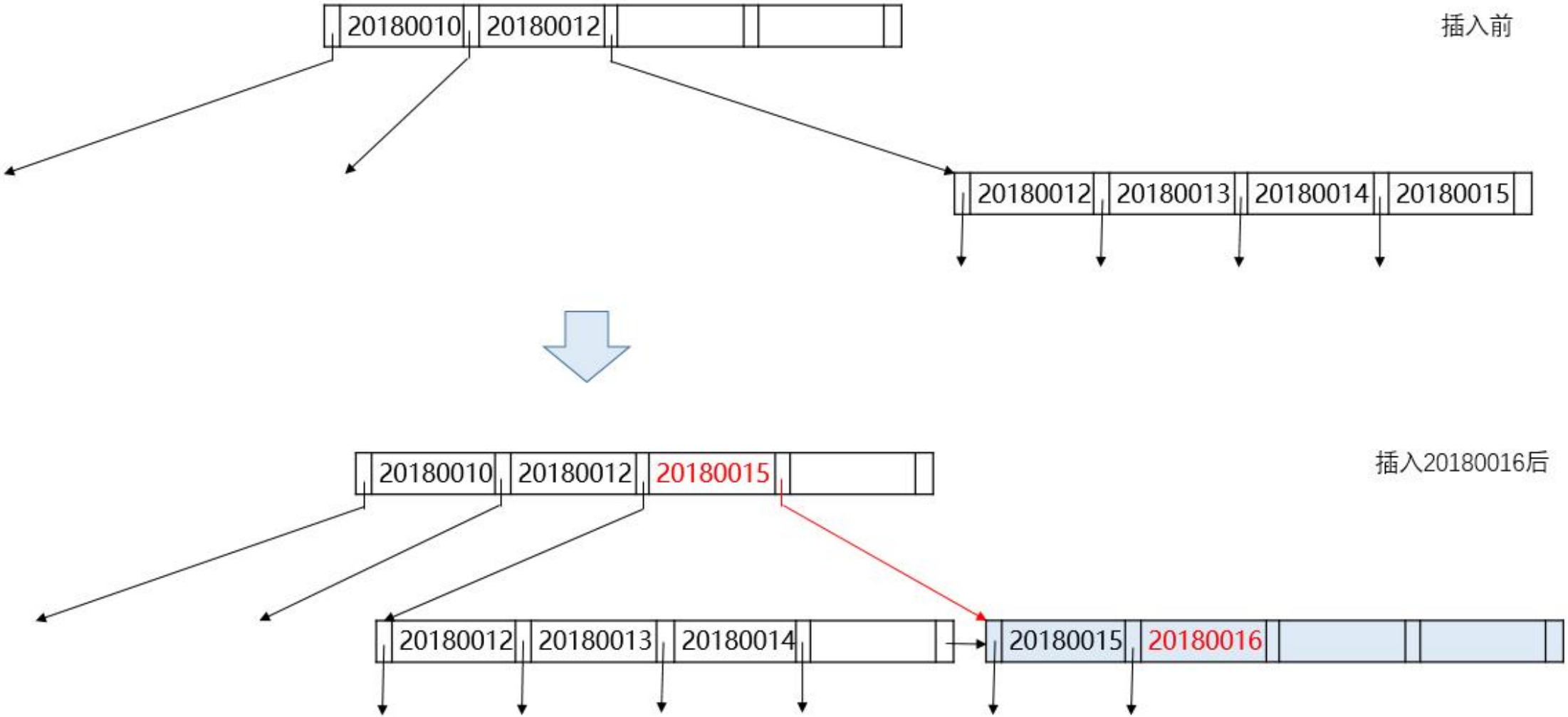
三、B+树索引的维护

插入元组

- 叶结点有空闲空间（Key值个数小于n-1）
  - 直接插入
- 叶结点达到最大充满度（key值个数等于n-1）
  - 当前叶结点分裂成2个叶结点
  - 父结点插入
  - 维护父结点
  - 逐层向上，直到插入完成

若根结点分裂会导致树的高度增加一层

三、B+树索引的维护


20180010
20180012
插入前
20180012
20180013
20180014
20180015
20180010
20180012
20180015
插入20180016后
20180012
20180013
20180014
20180015
20180016

三、B+树索引的维护

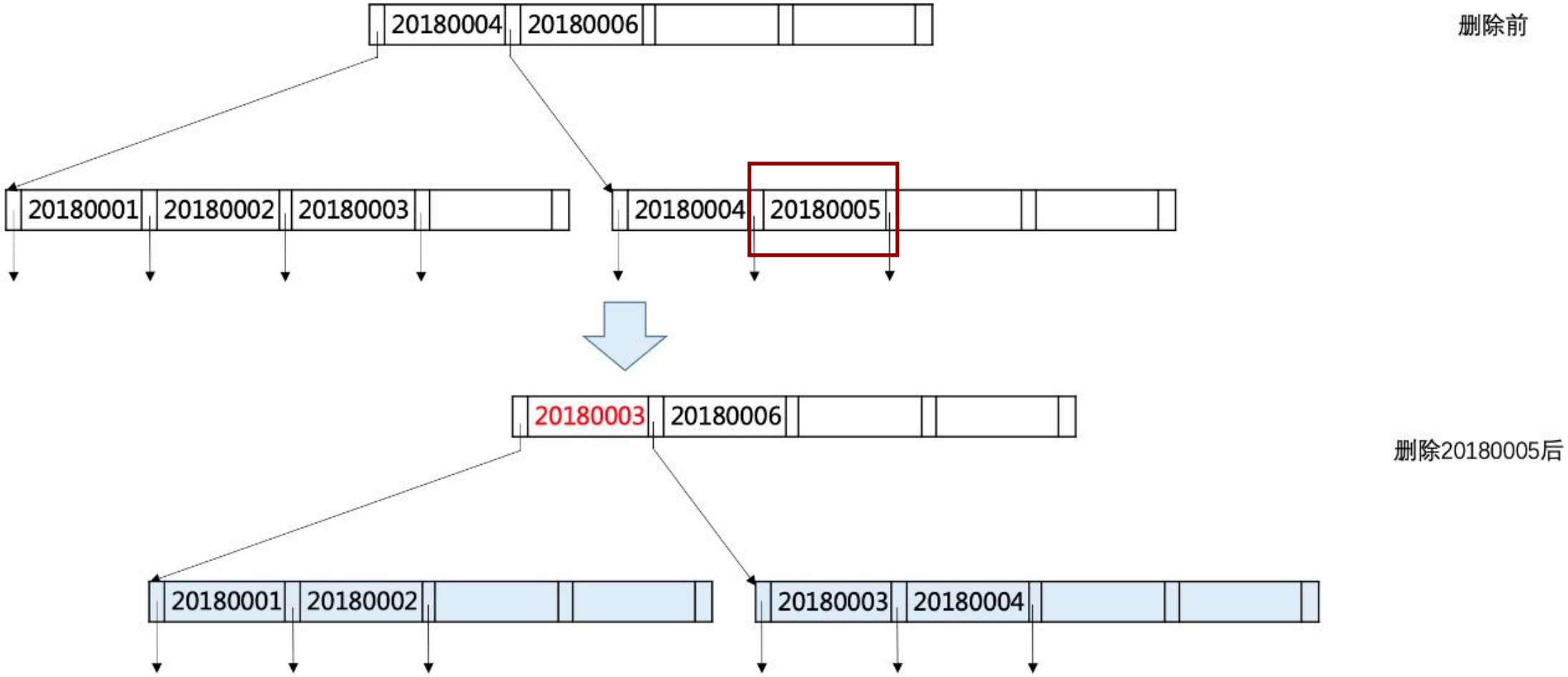
- 删除元组
  - 用随机搜索算法找到要删除的索引项
  - 删除后叶结点仍能满足最小充满度（Key值个数大于 ⌈(n-1)/2⌉ ）
    - 直接删除

三、B+树索引的维护

- 删除元组
  - 删除后叶结点不能满足最小充满度（Key值个数小于等于⌈(n-1)/2⌉）
    - 删除当前索引项
    - 合并叶结点
    - 删除父结点索引项
    - 维护父结点
    - 逐层向上，直到删除完成

若合并传递至根结点会导致树的高度降低一层

三、B+树索引的维护



删除前

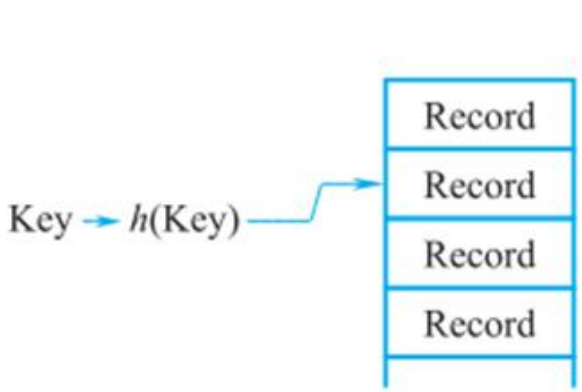
删除20180005后

04

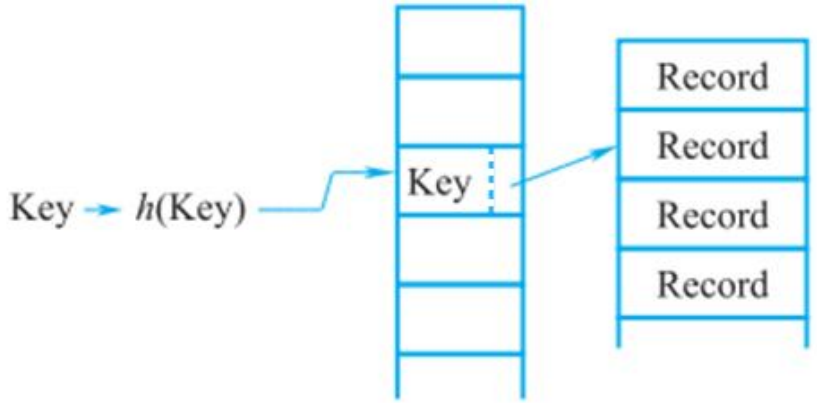
哈希索引

哈希索引

- 两个关键要素：
  - 哈希表
    - B个哈希桶
  - 哈希函数
    - 将记录的索引属性值影射到哈希桶
    - 每个桶存放放一条或多条哈希值相同的索引项
    - 每个索引项包括属性值和指向相应记录的指针



(a) 哈希存储方式



(b) 哈希索引

哈希文件组织中，哈希桶中存放的数据记录

一、静态哈希索引

- 静态哈希索引
  - 哈希函数
  - 哈希表
  - 哈希索引的查找及维护

一、静态哈希索引

1、哈希函数的设计原则

- 尽量保证函数的取值随机和均匀
  - 不会出现某个桶的索引项远远超过其他桶

例：在Student表的Sno属性上建哈希索引（5000条元组，100个桶）

- 学号转换为数值，除以100取模，以此为桶号；
- 学号转换为数值，取学号后两位作为桶号；

一、静态哈希索引

2、哈希表的构成：

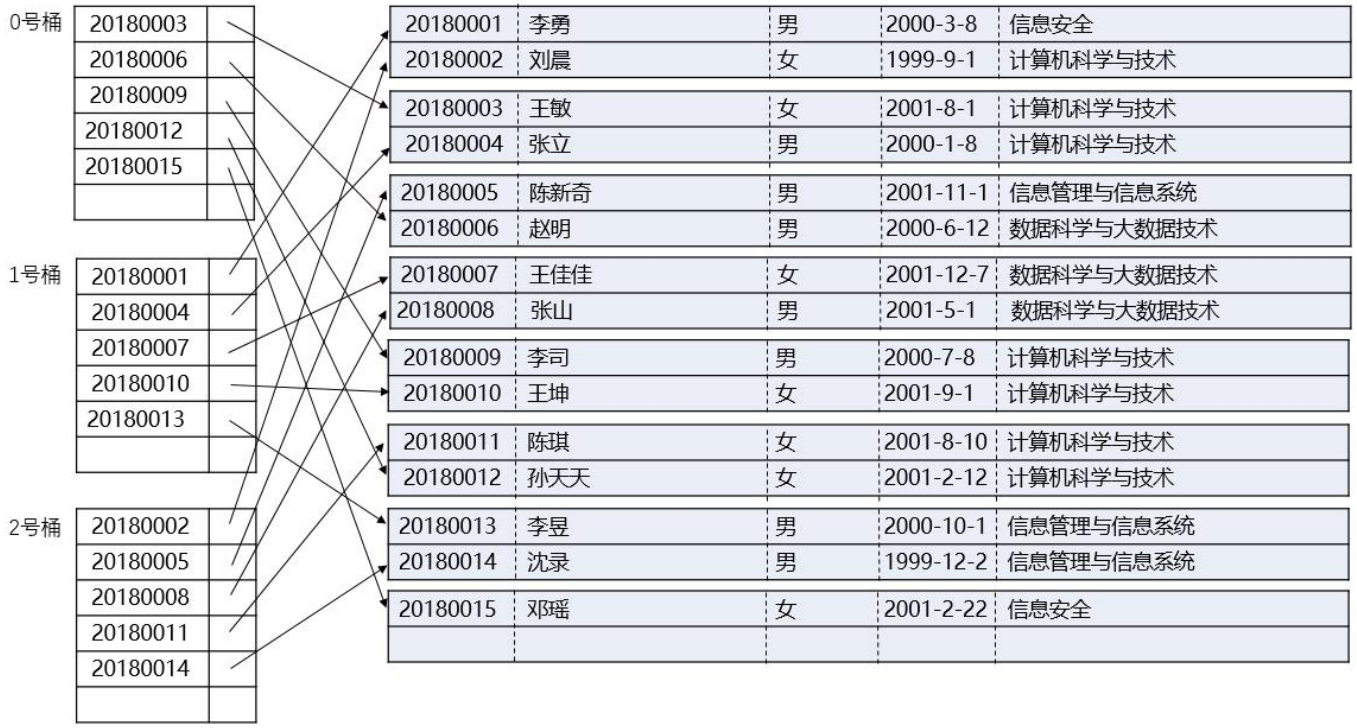
- 由一组桶组成，一个桶对应一个或多个物理块；
- 桶中存放被映射到该桶的哈希索引项；
- 每条索引项包括索引属性值和指向相应记录的指针；

一、静态哈希索引

2、哈希表的构成：

例：Student表Sno属性上的哈希索引

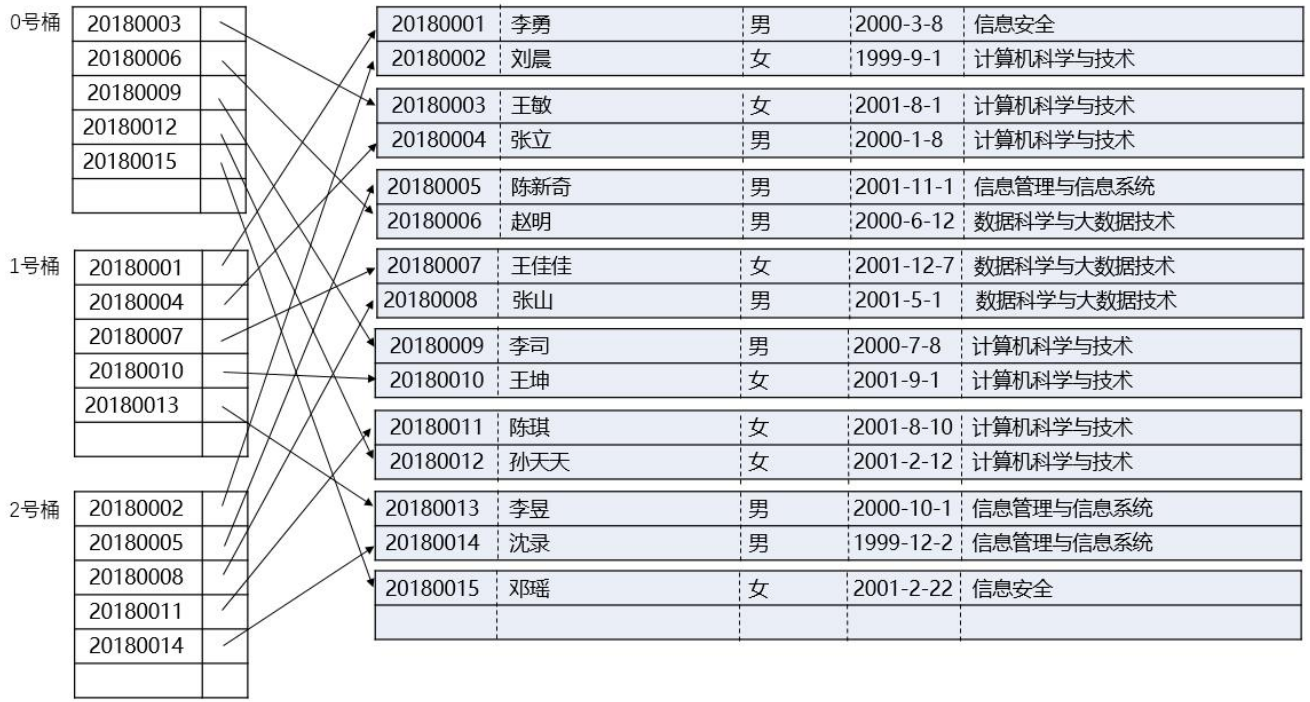
- 每个桶最多可以存放6个索引项，共15条数据，设有3个哈希桶。
- 哈希函数是取学号最后2位，将其转换数值型，除以3并取模作为桶号



一、静态哈希索引

- 桶溢出：某个桶的存储空间不足
- 桶溢出的三类主要原因：
  - 哈希桶数量不足，不能存放所有的索引项

例：下图中元组数量超过18条



一、静态哈希索引

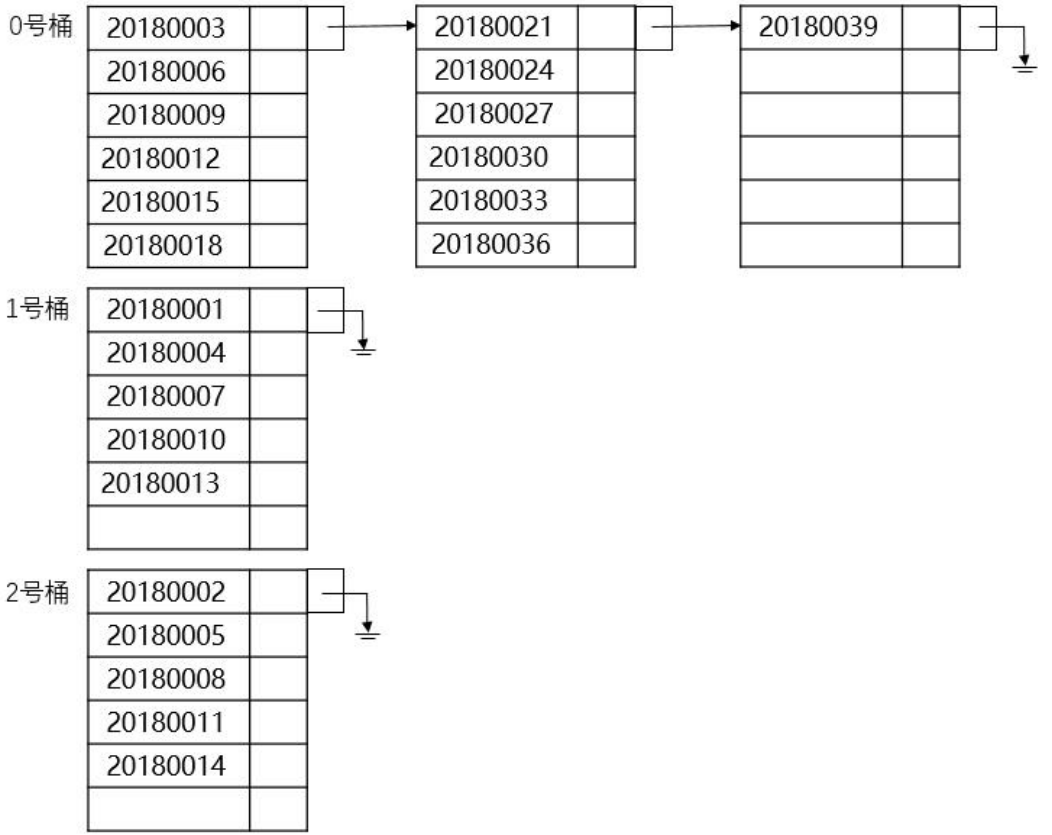
- 桶溢出的三类主要原因：
  - 属性存在偏斜，某些属性值过多。

    例：Student表中，‘计算机科学与技术’专业的学生人数非常多，使得相应的桶无法容纳下所有的索引项

  - 哈希函数设计不合理，无法将索引均匀地映射到每个桶，导致某个桶的数据过多。

一、静态哈希索引

- 减少桶溢出的措施：
  - 预留一定百分比的空间
    - 溢出仍不可避免
  - 分配溢出桶
    - 当溢出桶空间充满时，追加新的溢出桶
    - 用溢出链将一个桶及其溢出桶链接在一起



一、静态哈希索引

3、哈希索引的查找及维护

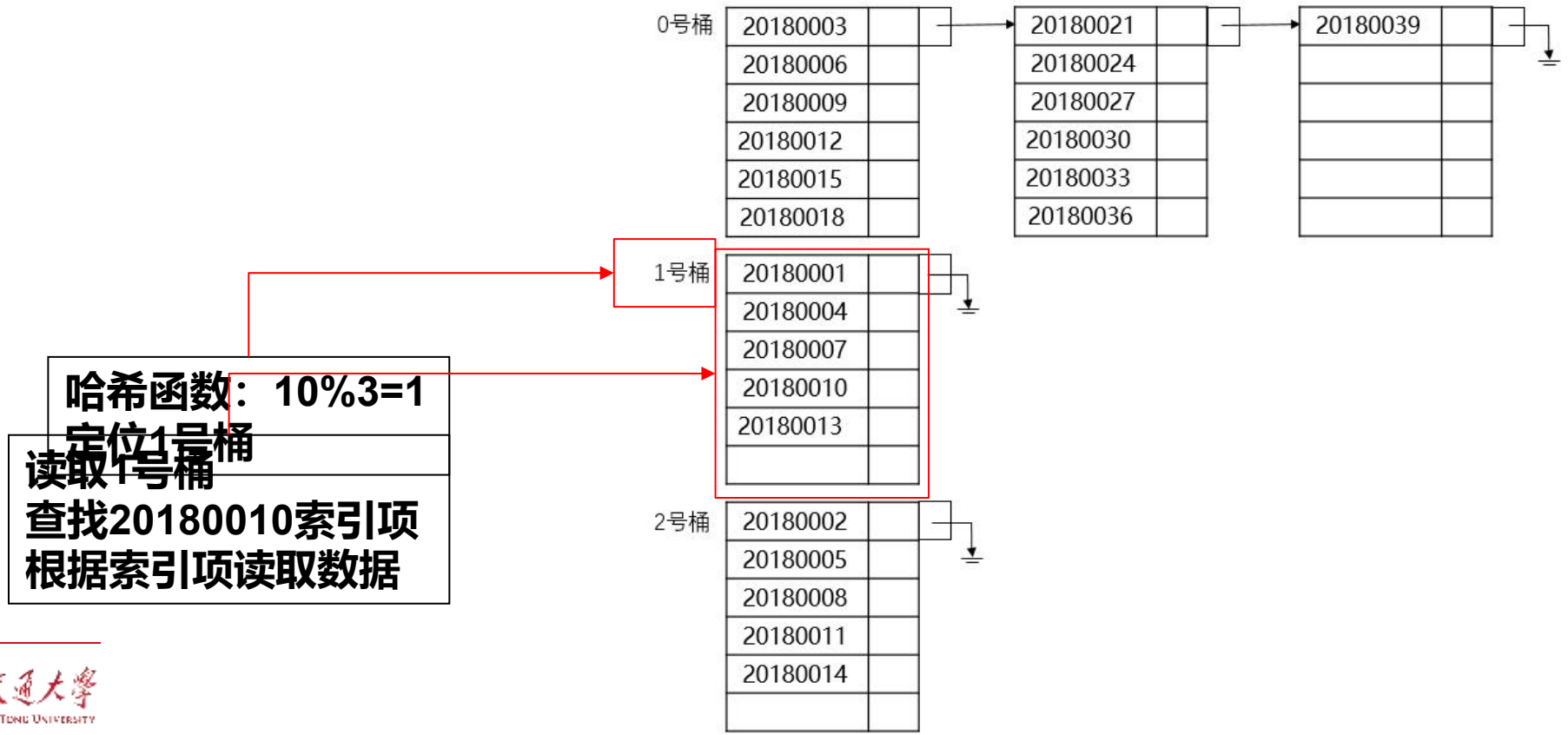
1）哈希索引的查找；

- 擅长做等值查找；
  - 根据属性值计算出哈希函数值，得到桶号；
  - 到相应的桶中搜索相应的索引项；
  - 若桶中没有找到，但存在溢出桶，继续搜索溢出桶；

一、静态哈希索引

1）哈希索引的查找；

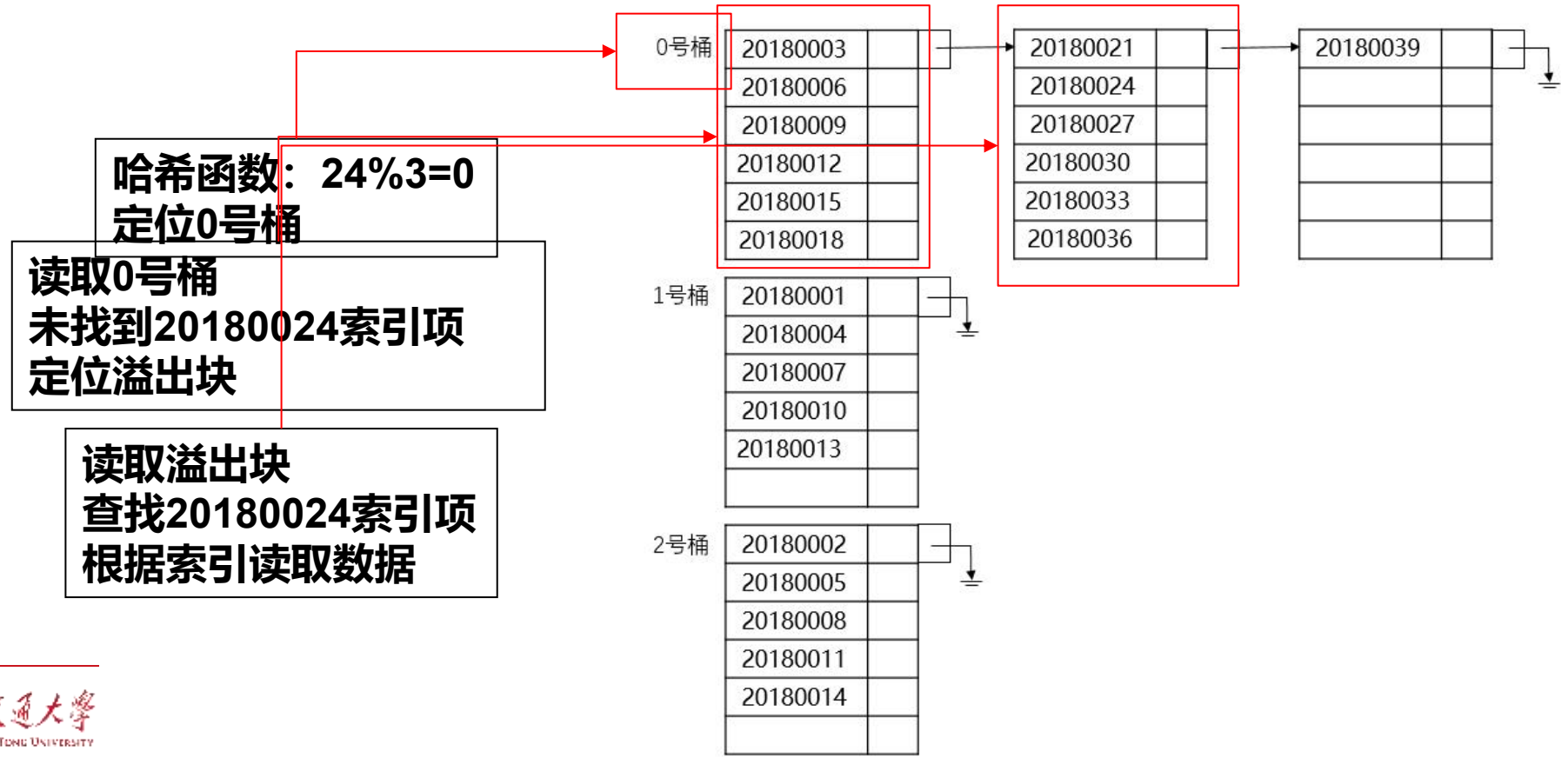
例：查找学号为20180010的学生



一、静态哈希索引

1）哈希索引的查找；

例：查找学号为20180024的学生



一、静态哈希索引

3、哈希索引的查找及维护

2）哈希索引的维护；

■插入元组时，需向哈希索引中插入相应的索引项；

●哈希函数计算桶号

✓若桶中有空间，直接插入

✓若桶中空间已满，申请溢出桶并插入

一、静态哈希索引

3、哈希索引的查找及维护

2）哈希索引的维护；

■删除元组时，需删除相应的索引项；

●哈希函数计算桶号

✓若没有溢出块，直接删除；

✓若有溢出块，判断是否有空间合并溢出块；

二、动态哈希索引

- 静态哈希索引中，桶的个数事先确定且不再改变
  - 桶溢出不可避免，影响查找效率
- 两种解决方法：
  - 适当预留桶空间
    - 表数据量难以估算
  - 随着关系表的增大，周期性地增加哈希桶数目，并重组索引
    - 耗时，影响正在进行的查询

二、动态哈希索引

动态哈希索引

- 随着关系表的增大，逐渐扩大桶的数目
- 两类动态哈希索引：
  - 可扩展哈希索引
  - 线性哈希索引

05

位图索引

位图索引

位图索引

- 长度为n的位向量集合
  - n：索引属性的基数（可能取值的个数）
  - 每一个位向量对应于索引属性的一个可能的取值
- 如果第i条记录的索引属性值为v，那么对应于值v的位向量在位置i上取值为1，其他的位向量在位置i上取值为0。

位图索引

例：在Student表的Ssex属性上建立位图索引

- 性别属性Ssex只有“男”和“女”两个可能的取值，基数为2，因而Ssex属性上的位图索引为长度为2的位向量集合

Ssno	Sname	Ssex	Sbirthdate	Smajor	男	女
20180001	李勇	男	2000-3-8	信息安全	1	0
20180002	刘晨	女	1999-9-1	计算机科学与技术	0	1
20180003	王敏	女	2001-8-1	计算机科学与技术	0	1
20180004	张立	男	2000-1-8	计算机科学与技术	1	0
20180005	陈新奇	男	2001-11-1	信息管理与信息系统	1	0
20180006	赵明	男	2000-6-12	数据科学与大数据技术	1	0
20180007	王佳佳	女	2001-12-7	数据科学与大数据技术	0	1
20180008	张山	男	2001-5-1	数据科学与大数据技术	1	0
20180009	李司	男	2000-7-8	计算机科学与技术	1	0
20180010	王坤	女	2001-9-1	计算机科学与技术	0	1
20180011	陈琪	女	2001-8-10	计算机科学与技术	0	1
20180012	孙天天	女	2001-2-12	计算机科学与技术	0	1
20180013	李昱	男	2000-10-1	信息管理与信息系统	1	0
20180014	沈录	男	1999-12-2	信息管理与信息系统	1	0
20180015	邓瑶	女	2001-2-22	信息安全	0	1

10对应男生

01对应女生

位图索引

位图索引

- 例：查询Student表的所有女生

```
SELECT * FROM Student
WHERE Ssex='女';
```

借助Ssex属性上的位图索引，取出第2位为1的所有元组。

- 例：统计Student表上男生和女生的人数

```
SELECT Ssex, Count(*)
FROM Student
GROUP By Ssex;
```

分别统计Ssex属性上位图索引各个位中1的个数（无需访问基本表）。

位图索引

例：统计计算机科学与技术专业男生人数

```
SELECT count(*)
FROM Student
WHERE Ssex='男' AND Smajor='计算机科学与技术';
```

- 将Ssex属性位图索引的第1位与Smajor属性位图索引的第2位进行与操作，得到结果向量：

(0，0，0,1，0，0，0，0，1，0，0，0，0，0，0)

- 对结果向量中1的个数进行计数，结果为2

位图索引

例：统计计算机科学与技术专业男生人数

与运算：(0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0)

Ssno	Sname	Ssex	Sbirthdate	Smajor
20180001	李勇	男	2000–3–8	信息安全
20180002	刘晨	女	1999–9–1	计算机科学与技术
20180003	王敏	女	2001–8–1	计算机科学与技术
20180004	张立	男	2000–1–8	计算机科学与技术
20180005	陈新奇	男	2001–11–1	信息管理与信息系统
20180006	赵明	男	2000–6–12	数据科学与大数据技术
20180007	王佳佳	女	2001–12–7	数据科学与大数据技术
20180008	张山	男	2001–5–1	数据科学与大数据技术
20180009	李司	男	2000–7–8	计算机科学与技术
20180010	王坤	女	2001–9–1	计算机科学与技术
20180011	陈琪	女	2001–8–10	计算机科学与技术
20180012	孙天天	女	2001–2–12	计算机科学与技术
20180013	李昱	男	2000–10–1	信息管理与信息系统
20180014	沈录	男	1999–12–2	信息管理与信息系统
20180015	邓瑶	女	2001–2–22	信息安全

男	女
1	0
0	1
0	1
1	0
1	0
1	0
0	1
1	0
1	0
0	1
0	1
0	1
1	0
1	0
0	1

Ssno	Sname	Ssex	Sbirthdate	Smajor
20180001	李勇	男	2000–3–8	信息安全
20180002	刘晨	女	1999–9–1	计算机科学与技术
20180003	王敏	女	2001–8–1	计算机科学与技术
20180004	张立	男	2000–1–8	计算机科学与技术
20180005	陈新奇	男	2001–11–1	信息管理与信息系统
20180006	赵明	男	2000–6–12	数据科学与大数据技术
20180007	王佳佳	女	2001–12–7	数据科学与大数据技术
20180008	张山	男	2001–5–1	数据科学与大数据技术
20180009	李司	男	2000–7–8	计算机科学与技术
20180010	王坤	女	2001–9–1	计算机科学与技术
20180011	陈琪	女	2001–8–10	计算机科学与技术
20180012	孙天天	女	2001–2–12	计算机科学与技术
20180013	李昱	男	2000–10–1	信息管理与信息系统
20180014	沈录	男	1999–12–2	信息管理与信息系统
20180015	邓瑶	女	2001–2–22	信息安全

信息安全	计算机科学与技术	信息管理与信息系统	数据科学与大数据技术
1	0	0	0
0	1	0	0
0	1	0	0
0	1	0	0
0	0	1	0
0	0	0	1
0	0	0	1
0	0	0	1
0	1	0	0
0	1	0	0
0	1	0	0
0	1	0	0
0	0	1	0
0	0	1	0
1	0	0	0

位图索引

- 位图索引
  - 有效处理多值查询
  - 例：

```
SELECT *
FROM Student
WHERE Smajor in (‘数据科学与大数据技术’, ’信息安全’);
```

对Smajor属性的位图索引的第1位和第4位进行或操作，得到结果向量：

（1，0，0，0，1，0，0，0，0，0，0，0，1，1，1）

位图索引

或运算： (1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1)

Ssno	Sname	Ssex	Sbirthdate	Smajor
20180001	李勇	男	2000–3–8	信息安全
20180002	刘晨	女	1999–9–1	计算机科学与技术
20180003	王敏	女	2001–8–1	计算机科学与技术
20180004	张立	男	2000–1–8	计算机科学与技术
20180005	陈新奇	男	2001–11–1	信息管理与信息系统
20180006	赵明	男	2000–6–12	数据科学与大数据技术
20180007	王佳佳	女	2001–12–7	数据科学与大数据技术
20180008	张山	男	2001–5–1	数据科学与大数据技术
20180009	李司	男	2000–7–8	计算机科学与技术
20180010	王坤	女	2001–9–1	计算机科学与技术
20180011	陈琪	女	2001–8–10	计算机科学与技术
20180012	孙天天	女	2001–2–12	计算机科学与技术
20180013	李昱	男	2000–10–1	信息管理与信息系统
20180014	沈录	男	1999–12–2	信息管理与信息系统
20180015	邓瑶	女	2001–2–22	信息安全

信息安全	计算机科学与技术	信息管理与信息系统	数据科学与大数据技术
1	0	0	0
0	1	0	0
0	1	0	0
0	1	0	0
0	0	1	0
0	0	0	1
0	0	0	1
0	0	0	1
0	1	0	0
0	1	0	0
0	1	0	0
0	1	0	0
0	0	1	0
0	0	1	0
1	0	0	0

位图索引

位图索引的优劣

优势：

- 使用位操作来快速回答用户查询或快速定位满足条件的元组集合，减少对基本表的全表扫描，提高查询效率。

劣势：

- 位图索引的大小与列的基数成正比，基数大的列其位图索引会非常庞大，因此它只适用于基数小的属性列。

位图索引

- 编码位图索引（Encoded Bitmap Index ）
  - 改进标准位图索引
    - 通过对属性值编码，减少索引位向量的个数，从而能够应对有较高基数的列。

Ssno	Sname	Ssex	Sbirthdate	Smajor		
20180001	李勇	男	2000-3-8	信息安全	0	0
20180002	刘晨	女	1999-9-1	计算机科学与技术	0	1
20180003	王敏	女	2001-8-1	计算机科学与技术	0	1
20180004	张立	男	2000-1-8	计算机科学与技术	0	1
20180005	陈新奇	男	2001-11-1	信息管理与信息系统	1	0
20180006	赵明	男	2000-6-12	数据科学与大数据技术	1	1
20180007	王佳佳	女	2001-12-7	数据科学与大数据技术	1	1
20180008	张山	男	2001-5-1	数据科学与大数据技术	1	1
20180009	李司	男	2000-7-8	计算机科学与技术	0	1
20180010	王坤	女	2001-9-1	计算机科学与技术	0	1
20180011	陈琪	女	2001-8-10	计算机科学与技术	0	1
20180012	孙天天	女	2001-2-12	计算机科学与技术	0	1
20180013	李昱	男	2000-10-1	信息管理与信息系统	1	0
20180014	沈录	男	1999-12-2	信息管理与信息系统	1	0
20180015	邓瑶	女	2001-2-22	信息安全	0	0

编码映射表

信息安全	00
计算机科学与技术	01
信息管理与信息系统	10
数据科学与大数据技术	11

位图索引

编码位图索引的优劣

- 优势：减少索引位个数
  - 如果索引属性列的基数为K，标准位图索引所需要的位向量个数为K个，而编码位图索引所需要的位向量个数仅为$\log_2 K$个 。
- 劣势：查询时需要访问所有位向量才能完成
  - 例：在学生表中查找信息管理与信息系统专业和信息安全专业的所有学生，需要扫描所有位向量，从中查找编码为10或00的元组

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

THANK YOU !

饮水思源 爱国荣校