2024年春季学期

数据库系统概论

An Introduction to Database Systems

第七章　数据库设计

中国科学技术大学 大数据学院

黄振亚，huangzhy@ustc.edu.cn

http://staff.ustc.edu.cn/~huangzhy/Course/DB2024.html

第七章 数据库设计

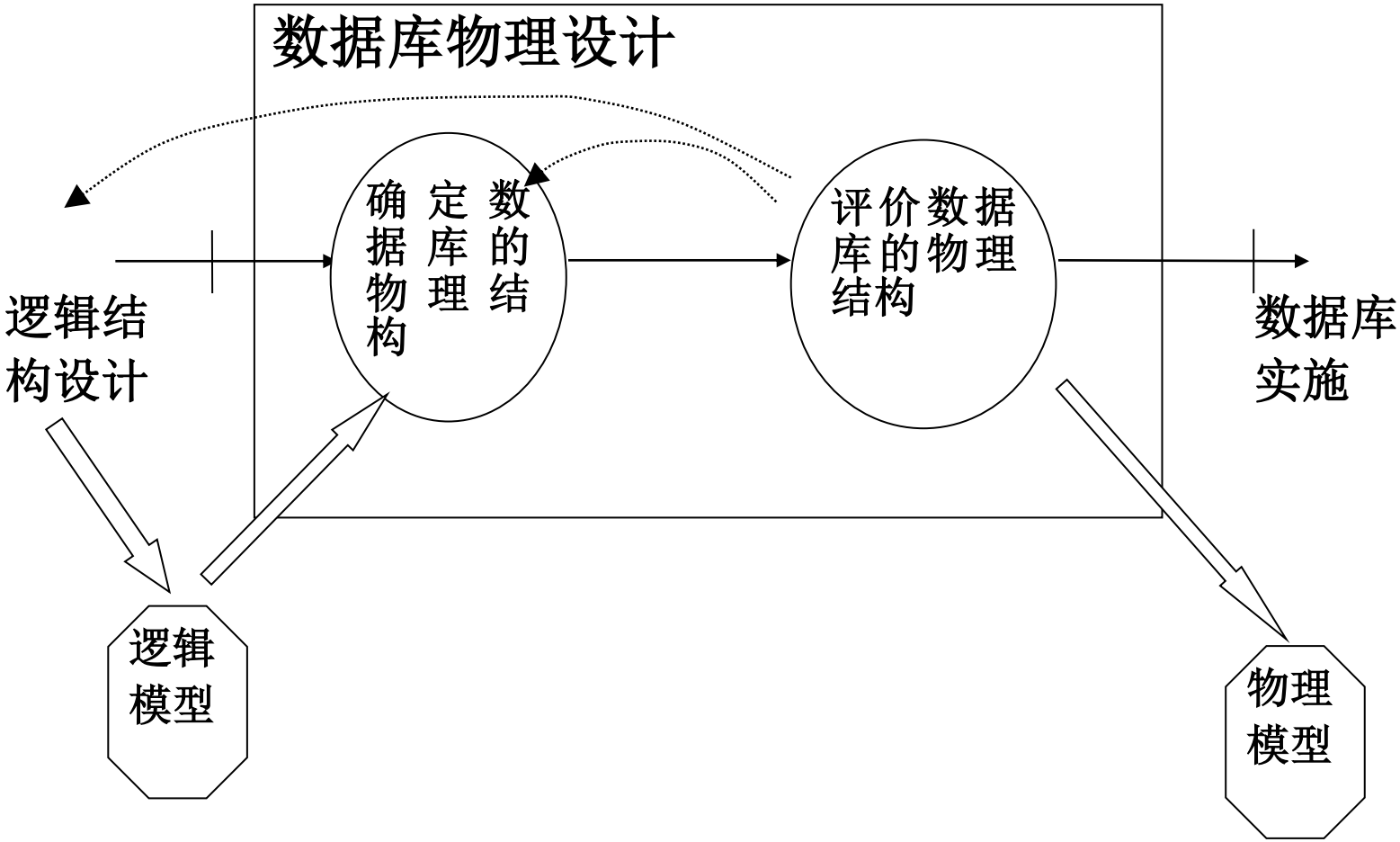
7.5 数据库的物理设计

- 数据库的物理设计
  - 数据库在物理设备上的存储结构与存取方法称为数据库的物理结构，它依赖于选定的数据库管理系统
  - 为一个给定的逻辑数据模型选取一个最适合应用环境的物理结构的过程，就是数据库的物理设计

数据库的物理设计(续)

- 数据库物理设计的步骤
  - 1. 确定数据库的物理结构
    - 在关系数据库中主要指存取方法和存储结构;
  - 2. 对物理结构进行评价
    - 评价的重点是时间和空间效率
  - 若评价结果满足原设计要求，则可进入到物理实施阶段。否则，就需要重新设计或修改物理结构，有时甚至要返回逻辑设计阶段修改数据模型。

数据库的物理设计(续)



回顾：数据字典

- **4. 数据存储**
  - 输入输出
  - 数据量
  - 存取频度：每小时、每天或每周存取次数，每次存取的数据量等信息
  - 存取方法：批处理 / 联机处理；检索 / 更新；顺序检索 / 随机检索
- **5. 处理过程**
  - 处理要求：处理频度要求，如单位时间里处理多少事务，多少数据量、响应时间要求等

物理设计的输入及性能评价的标准

7.5 数据库的物理设计

7.5.1 数据库物理设计的内容和方法

7.5.2 关系模式存取方法选择

7.5.3 确定数据库的存储结构

7.5.4 评价物理结构

7.5.1 数据库物理设计的内容和方法

- 设计物理数据库结构的准备工作
  - 充分了解应用环境，对要运行的事务进行详细分析，获得选择物理数据库设计所需参数
  - 充分了解所用RDBMS的内部特征，特别是系统提供的存取方法和存储结构
    - 有哪些索引(B+树，HASH，聚簇)，如何建立索引
    - 有哪些存储结构（行存储，列存储，块存储），如何选择

数据库的物理设计的内容和方法（续）

- 选择物理数据库设计所需参数
  - 数据库查询事务
    - 查询的关系
    - 查询条件所涉及的属性
    - 连接条件所涉及的属性
    - 查询的投影属性
  - 数据更新事务
    - 被更新的关系
    - 每个关系上的更新操作条件所涉及的属性
    - 修改操作要改变的属性值
  - 每个事务在各关系上运行的频率和性能要求

数据库的物理设计的内容和方法（续）

- 关系数据库物理设计的内容
  - 为关系模式选择存取方法(建立存取路径)
  - 设计关系、索引等数据库文件的物理存储结构

7.5 数据库的物理设计

7.5.1 数据库物理设计的内容和方法

7.5.2 关系模式存取方法选择

7.5.3 确定数据库的存储结构

7.5.4 评价物理结构

7.5.2 关系模式存取方法选择

- 数据库系统是多用户共享的系统，对同一个关系要建立多条存取路径才能满足多用户的多种应用要求
- 物理设计的任务之一就是要确定选择哪些存取方法，即建立哪些存取路径

建立索引

- 语句格式

CREATE [UNIQUE] INDEX <索引名>

ON <表名>

[USING 索引方法]（列名1，列名2，[, …]）；

■ **UNIQUE：此索引的每一个索引值只对应唯一的数据记录**

例：

CREATE UNIQUE INDEX Studentname ON Student USING Hash(sname);

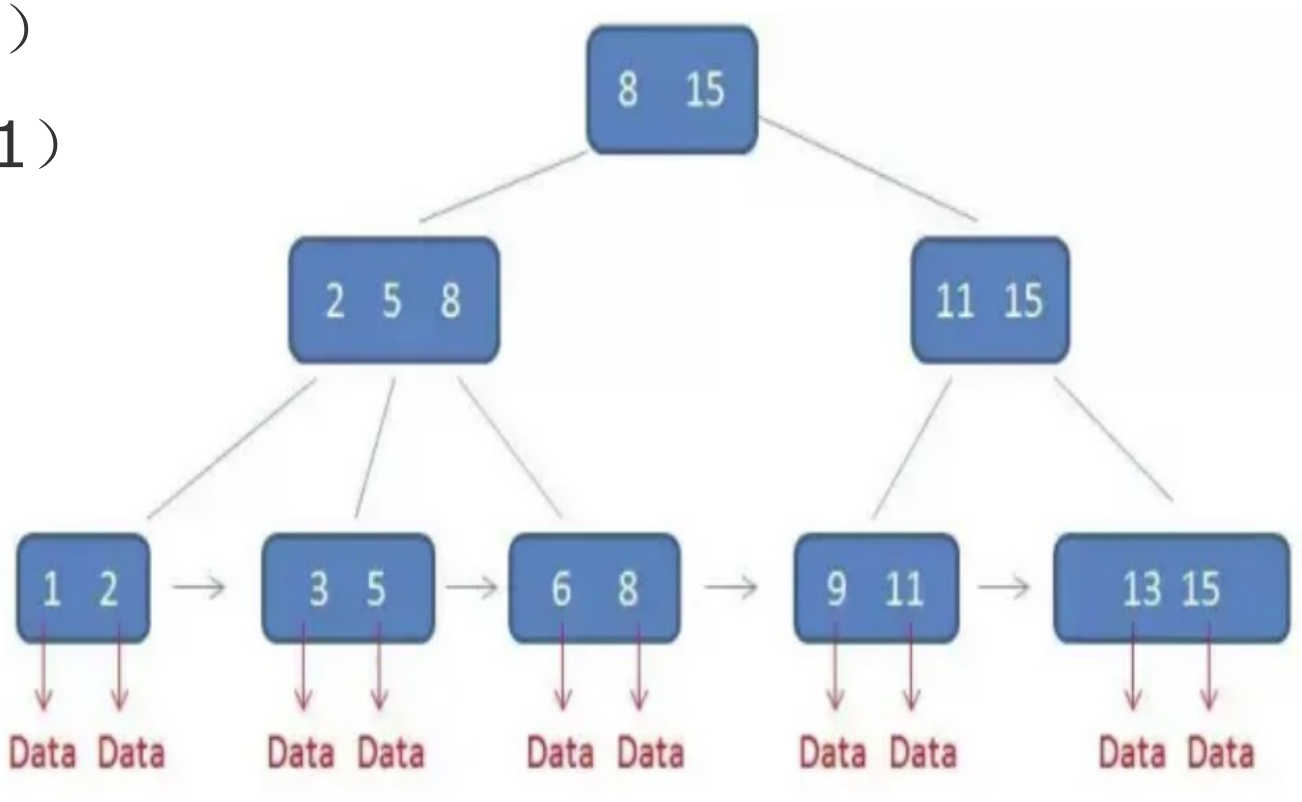
关系模式存取方法选择（续）

- DBMS常用存取方法
  - B+树索引方法
    - 经典存取方法，使用最普遍
  - HASH方法
  - 聚簇（Cluster）方法

B+树索引方法

B+树

- 索引与数据
- 单值查询（369）
- 范围查询（3-11）



一、索引存取方法的选择(B+树)

- 根据应用要求确定
  - 对哪些属性列建立索引
  - 对哪些属性列建立组合索引
  - 对哪些索引要设计为唯一索引

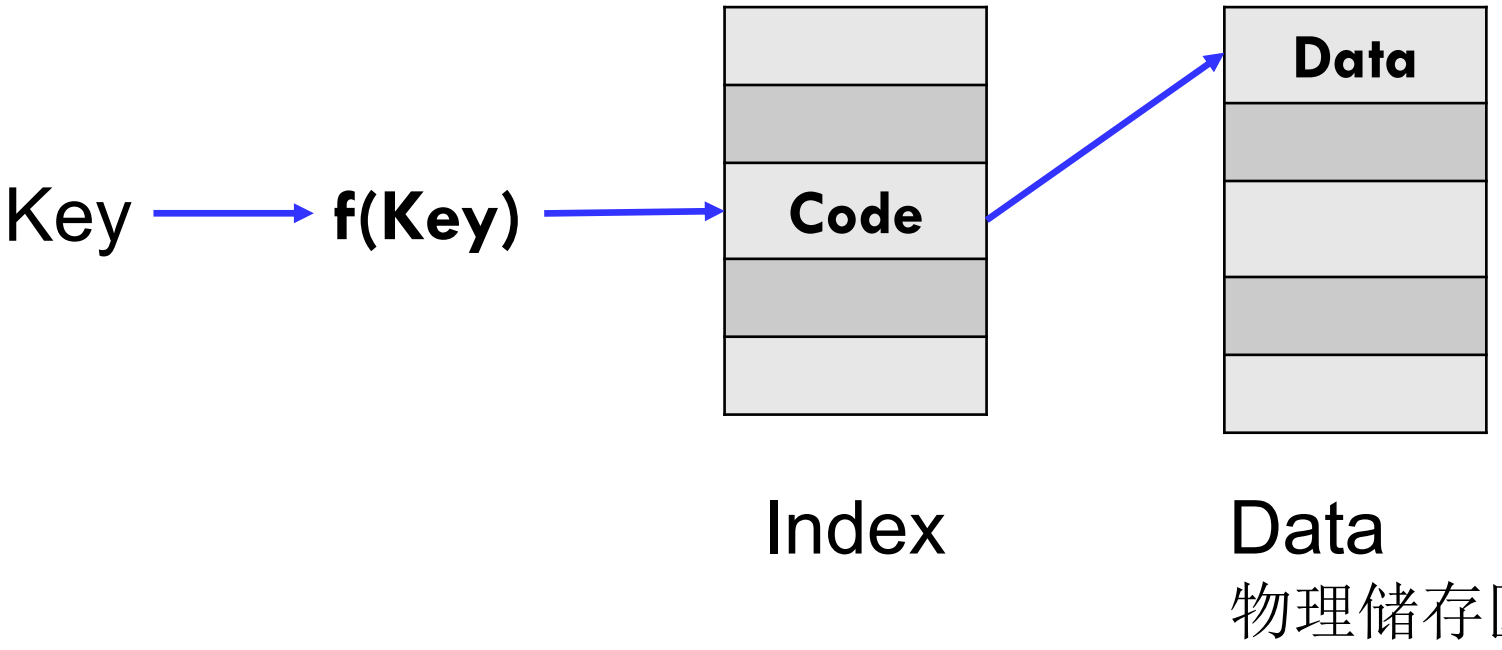
B+树索引存取方法的选择

- 选择索引存取方法的一般规则
  - 如果一个(或一组)属性经常在查询条件中出现，则考虑在这个(或这组)属性上建立索引(或组合索引)
  - 如果一个属性经常作为最大值和最小值等聚集函数的参数，则考虑在这个属性上建立索引
  - 如果一个(或一组)属性经常在连接操作的连接条件中出现，则考虑在这个(或这组)属性上建立索引
- 关系上定义的索引数过多会带来较多的额外开销
  - 维护索引的开销
  - 查找索引的开销

HASH索引

HASH

哈希函数 *f* (key)



二、HASH存取方法的选择

- 选择HASH存取方法的规则
  - 当一个关系满足下列两个条件时，可以选择HASH存取方法
    - 该关系的属性主要出现在等值连接条件中或主要出现在等值比较选择条件中
    - 该关系的大小可预知，而且不变
    - 该关系的大小动态改变，但所选用的DBMS提供了动态HASH存取方法

三、聚簇存取方法的选择

- 聚簇
  - 为了提高某个属性（或属性组）的查询速度，把这个或这些属性（称为聚簇码）上具有相同值的元组集中存放在连续的物理块称为聚簇
    - 该属性（或属性组）称为聚簇码（cluster key）
    - 许多关系型数据库管理系统都提供了聚簇功能
    - 聚簇存放与聚簇索引的区别

聚簇存取方法的选择（续）

- 聚簇索引
  - 建立聚簇索引后，基表中数据也需要按指定的聚簇属性值的升序或降序存放。也即聚簇索引的索引项顺序与表中元组的物理顺序一致。
  - 在一个基本表上最多只能建立一个聚簇索引
- 聚簇索引的适用条件
  - 很少对基表进行增删操作
  - 很少对其中的变长列进行修改操作

建立索引

- 语句格式

CREATE [CLUSTER] INDEX <索引名>

ON <表名> (<列名>)

- **CLUSTER：** 表示要建立的索引是聚簇索引

不同DBMS的实现方式不同

回顾：建立索引

[例]在Student表的Sage列上建立一个聚簇索引

```
CREATE CLUSTER INDEX Stuage
 ON  Student(Sage);
```

“年龄”属性建立聚簇索引，是合适的

[例] CREATE CLUSTER INDEX Stusname

```
 ON  Student(Sname);
```

在Student表的Sname（姓名）列上建立一个聚簇索引，而且Student表中的记录将按照Sname值的升序存放

聚簇存取方法的选择（续）

- 聚簇的用途
  - 1. 大大提高按聚簇码进行查询的效率

    例：学生关系按所在系建有索引，现查询DS系的学生名单。

    - 随机存放：DS系的500名学生分布在500个不同的物理块上时，至少要执行500次I/O操作（注意索引与数据地址）
    - 按照系名聚簇存放：将同一系的学生元组聚簇存放，则每读一个物理块可得到多个满足查询条件的元组，显著减少访问磁盘次数。DS系500名学生聚簇存放50个物理块，只要执行50次I/O
  - 2. 节省存储空间
    - 聚簇以后，聚簇码相同的元组集中在一起了，因而聚簇码值不必在每个元组中重复存储，只要在一组中存一次就行了

聚簇存取方法的选择（续）

- 聚簇的适用范围

1. 既适用于单个关系独立聚簇，也适用于多个关系组合聚簇

例：假设用户经常要按系别查询学生成绩单，

Select sname, cno, grade from student, sc where student.sno=sc.sno

- 查询涉及Student关系和SC关系的连接操作，按学号连接
  - 把具有相同学号值的学生元组和选修元组在物理上聚簇在一起。
  - 相当于把多个关系按“预连接”的形式存放，
  - 从而大大提高连接操作的效率。

聚簇存取方法的选择（续）

- 聚簇的适用范围

2. 当通过聚簇码进行访问或连接是该关系的主要应用，与聚簇码无关的其他访问很少或者是次要的时，可以使用聚簇

- **当SQL语句中包含有与聚簇码有关的ORDER BY，GROUP BY，UNION，DISTINCT等子句或短语时，使用聚簇特别有利，可以省去对结果集的排序操作**

聚簇存取方法的选择（续）

- 聚簇的局限性
  - 一个基表上最多建立一个聚簇索引
  - 聚簇只能提高某些特定应用的性能
  - 建立与维护聚簇的开销相当大
    - 对已有关系建立聚簇，将导致关系中元组移动其物理存储位置，并使此关系上原有的索引无效，必须重建
    - 当一个元组的聚簇码改变时，该元组的存储位置也要做相应移动
      - 例，学生从CS换到DS系

聚簇存取方法的选择（续）

- 聚簇的适用条件：设计候选聚簇
  - 经常在一起进行连接操作的关系可以建立聚簇
  - 一个关系的一组属性经常出现在相等比较条件中，则该单个关系可建立聚簇
  - 一个关系的一个(或一组)属性上的值重复率很高，则此单个关系可建立聚簇
    - 即对应每个聚簇码值的平均元组数不太少。聚簇的效果不明显

聚簇存取方法的选择（续）

- 聚簇的适用条件：优化聚簇设计

  （1）从聚簇中删除经常进行全表扫描的关系

  （2）从聚簇中删除更新操作远多于连接操作的关系

  （3）从聚簇中删除重复出现的关系

  - 不同的聚簇中可能包含相同的关系，一个关系可以在某一个聚簇中，但不能同时加入多个聚簇
    - 从这多个聚簇方案(包括不建立聚簇)中选择一个较优的，即在这个聚簇上运行各种事务的总代价最小

7.5 数据库的物理设计

7.5.1 数据库物理设计的内容和方法

7.5.2 关系模式存取方法选择

7.5.3 确定数据库的存储结构

7.5.4 评价物理结构

7.5.3 确定数据库的存储结构

- 确定数据库物理结构的内容
  - 1. 确定数据的存放位置和存储结构
    - 关系
    - 索引
    - 聚簇
    - 日志
    - 备份

    - 内存/磁盘
    - 列存放，行存储
    - 集中存放，分散存放
    - 顺序存放，随机存放，聚簇存放
  - 2. 确定系统配置

1. 确定数据的存放位置

- 确定数据存放位置和存储结构的因素
  - 存取时间
  - 存储空间利用率
  - 维护代价

  这三个方面常常是相互矛盾的

  例：消除一切冗余数据虽能够节约存储空间和减少维护代价，但往往会导致检索代价的增加

  必须进行权衡，选择一个折中方案

确定数据的存放位置（续）

- 基本原则
  - 根据应用情况将
    - 易变部分与稳定部分分开存放
    - 存取频率较高部分与存取频率较低部分，分开存放本
    - 日志与数据库对象（表，索引等）分开存放

确定数据的存放位置（续）

例：

- 数据库数据备份、日志文件备份等由于只在故障恢复时才使用，而且数据量很大，可以考虑存放在磁带上
- 如果计算机有多个磁盘或磁盘阵列，可以考虑将表和索引分别放在不同的磁盘上，在查询时，由于磁盘驱动器并行工作，可以提高物理I/O读写的效率
- 可以将比较大的表分别放在两个磁盘上，以加快存取速度，这在多用户环境下特别有效
- 可以将日志文件与数据库对象（表、索引等）放在不同的磁盘以改进系统的性能

2. 确定系统配置

- **DBMS产品一般都提供了一些存储分配参数**
  - 同时使用数据库的用户数
  - 同时打开的数据库对象数
  - 内存分配参数
  - 缓冲区分配参数（使用的缓冲区长度、个数）
  - 存储分配参数
  - 物理块的大小
  - 物理块装填因子
  - 时间片大小
  - 数据库的大小
  - 锁的数目等

2. 确定系统配置

- 系统都为这些变量赋予了合理的缺省值。

  在进行物理设计时需要根据应用环境确定这些参数值，以使系统性能最优。

- 在物理设计时对系统配置变量的调整只是初步的，要根据系统实际运行情况做进一步的调整，以切实改进系统性能。

7.5 数据库的物理设计

7.5.1 数据库物理设计的内容和方法

7.5.2 关系模式存取方法选择

7.5.3 确定数据库的存储结构

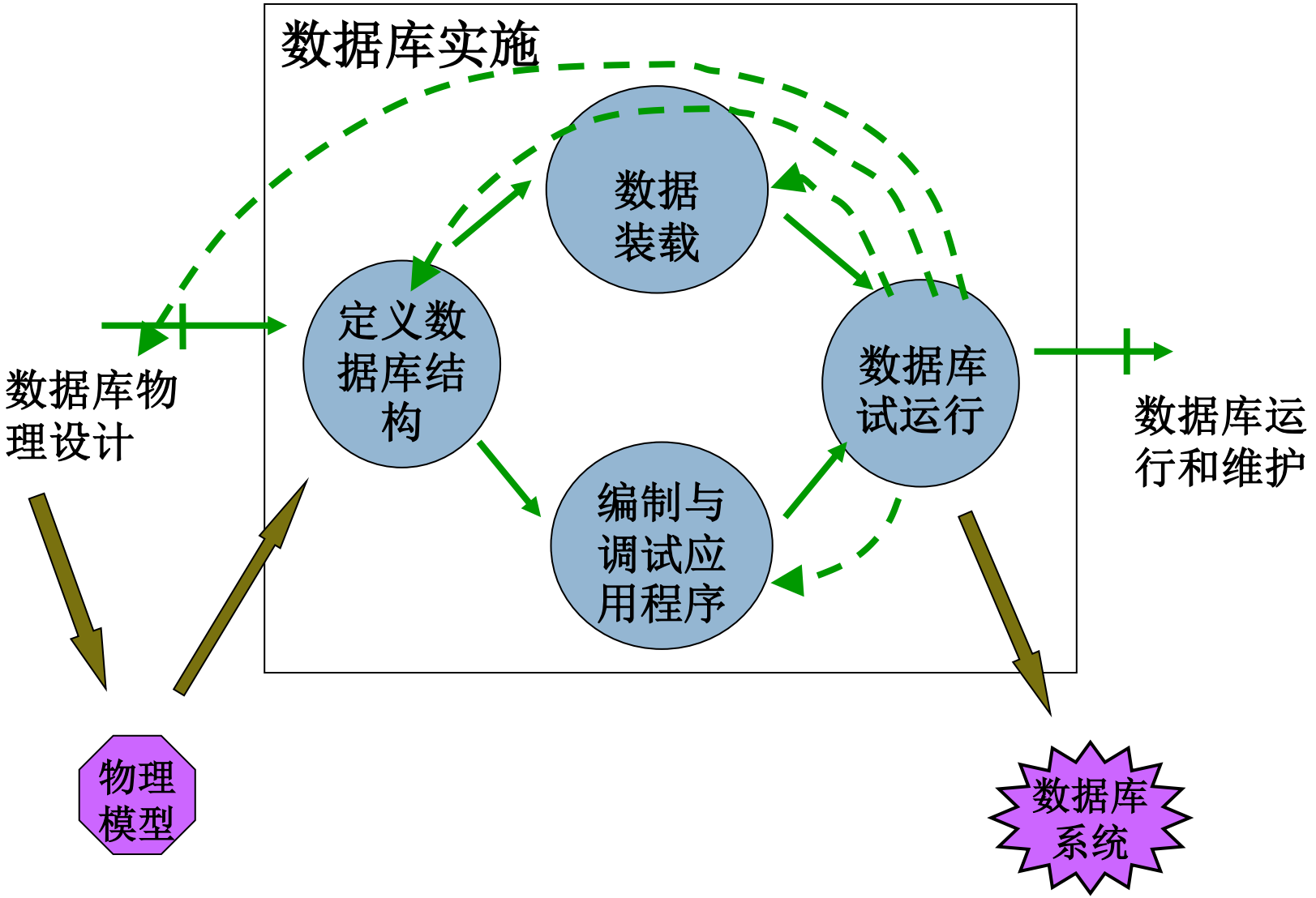
7.5.4 评价物理结构

7.5.4 评价物理结构

- 对数据库物理设计过程中产生的多种方案进行评价，从中选择一个较优的方案作为数据库的物理结构。
- 评价方法
  - 定量估算各种方案
    - 存储空间
    - 存取时间
    - 维护代价
  - 对估算结果进行权衡、比较，选择出一个较优的合理的物理结构
  - 如果该结构不符合用户需求，则需要修改设计

第七章 数据库设计

7.6数据库实施和维护


数据库实施
数据装载
定义数据库结构
数据库试运行
编制与调试应用程序
数据库物理设计
数据库运行和维护
物理模型
数据库系统

7.6数据库实施和维护

7.6.1 数据的载入和应用程序的调试

7.6.2 数据库的试运行

7.6.3 数据库的运行和维护

7.6.1 数据的载入和应用程序的调试

- 数据的载入
- 应用程序的编码和调试

数据的载入

- 数据库结构建立好后，就可以向数据库中装载数据了。组织数据入库是数据库实施阶段最主要的工作。
- 数据装载ETL
  - 数据抽取
  - 数据转换
  - 数据载入
- 使用ETL工具辅助完成

ETL工作是费时、费力的　　数据预处理是费时、费力的

应用程序的编码和调试

- 数据库应用程序的设计应该与数据设计并行进行
- 在组织数据入库的同时还要调试应用程序
- 软件工程：应用程序的设计、编码和调试的方法、步骤

7.6数据库实施和维护

7.6.1 数据的载入和应用程序的调试

7.6.2 数据库的试运行

7.6.3 数据库的运行和维护

7.6.2 数据库的试运行

- 在原有系统的数据有一小部分已输入数据库后，就可以开始对数据库系统进行联合调试，称为数据库的试运行
- 主要工作包括：

1）功能测试

- 实际运行数据库应用程序，执行对数据库的各种操作，测试应用程序的功能是否满足设计要求
- 如果不满足，对应用程序部分则要修改、调整，直到达到设计要求

2）性能测试

- 测量系统的性能指标，分析是否达到设计目标
- 如果测试的结果与设计目标不符，则要返回物理设计阶段，重新调整物理结构，修改系统参数，某些情况下甚至要返回逻辑设计阶段，修改逻辑结构

数据库的试运行（续）

- 数据库性能指标的测量
  - 数据库物理设计阶段在评价数据库结构估算时间、空间指标时，作了许多简化和假设，忽略了许多次要因素，因此结果必然很粗糙。
  - 数据库试运行则是要实际测量系统的各种性能指标（不仅是时间、空间指标），如果结果不符合设计目标，则需要返回物理设计阶段，调整物理结构，修改参数；有时甚至需要返回逻辑设计阶段，调整逻辑结构。

数据库的试运行（续）

强调两点：

1. 分期分批组织数据入库
   - 重新设计物理结构甚至逻辑结构，会导致数据重新入库
   - 由于数据入库工作量太大，费时、费力，所以应分期分批地组织数据入库
     - 先输入小批量数据供调试用
     - 待试运行基本合格后再大批量输入数据
     - 逐步增加数据量，逐步完成运行评价

数据库的试运行（续）

2. 数据库的转储和恢复（第十章）

- 在数据库试运行阶段，系统还不稳定，硬、软件故障随时都可能发生
- 系统的操作人员对新系统还不熟悉，误操作也不可避免
- 因此必须做好数据库的转储和恢复工作，尽量减少对数据库的破坏。

7.6 数据库实施和维护

7.6.1 数据的载入和应用程序的调试

7.6.2 数据库的试运行

7.6.3 数据库的运行和维护

7.6.3 数据库的运行与维护

- 数据库试运行合格后，数据库即可投入正式运行。
- 数据库投入运行标志着开发任务的基本完成和维护工作的开始
- 对数据库设计进行评价、调整、修改等维护工作是一个长期的任务，也是设计工作的继续和提高。
  - 应用环境在不断变化
  - 数据库运行过程中物理存储会不断变化

7.6.3 数据库的运行与维护

- 在数据库运行阶段，对数据库经常性的维护工作主要是由DBA完成的，包括：
  1. 数据库的转储和恢复
  2. 数据库的安全性、完整性控制
  3. 数据库性能的监督、分析和改进
  4. 数据库的重组织和重构造

7.6.3 数据库的运行与维护

- 在数据库运行阶段，对数据库经常性的维护工作主要是由数据库管理员完成的，包括：

1. 数据库的转储和恢复

  - 数据库管理员要针对不同的应用要求制定不同的转储计划，定期对数据库和日志文件进行备份。
  - 一旦发生介质故障，即利用数据库备份及日志文件备份，尽快将数据库恢复到某种一致性状态。

数据库的运行和维护

2. 数据库的安全性、完整性控制

- 初始定义
  - 数据库管理员根据用户的实际需要授予不同的操作权限
  - 根据应用环境定义不同的完整性约束条件
- 修改定义
  - 当应用环境发生变化，对安全性的要求也会发生变化，数据库管理员需要根据实际情况修改原有的安全性控制
  - 由于应用环境发生变化，数据库的完整性约束条件也会变化，也需要数据库管理员不断修正，以满足用户要求

数据库的运行和维护

3. 数据库性能的监督、分析和改进

- 在数据库运行过程中，数据库管理员必须监督系统运行，对监测数据进行分析，找出改进系统性能的方法。
  - 利用监测工具获取系统运行过程中一系列性能参数的值
  - 通过仔细分析这些数据，判断当前系统是否处于最佳运行状态
  - 如果不是，则需要通过调整某些参数来进一步改进数据库性能

数据库的运行和维护（续）

4. 数据库的重组织与重构造

（1）数据库的重组织

- 数据库运行一段时间后，由于记录的不断增、删、改，会使数据库的物理存储变坏，从而降低数据库存储空间的利用率和数据的存取效率，使数据库的性能下降。

数据库的运行与维护（续）

- 重组织的形式
  - 全部重组织
    - 索引重组、单表重组、表空间重组
  - 部分重组织
    - 只对频繁增、删的表进行重组织
  - 重组织的目标
    - 提高系统性能
- 重组织的工作
  - 按原设计要求
    - 重新安排存储位置
    - 回收垃圾
    - 减少指针链
  - 数据库的重组织不会改变原设计的数据逻辑结构和物理结构

数据库运行与维护（续）

（2）数据库重构造

实体，联系发生了变化，根据新环境调整数据库的模式和内模式

- 增加新的数据项
- 改变数据项的类型
- 改变数据库的容量
- 增加或删除索引
- 修改完整性约束条件

第七章 数据库设计

7.7 小结

- 数据库的设计过程
  - 需求分析
  - 概念结构设计
  - 逻辑结构设计
  - 物理设计
  - 实施和维护

小结（续）

- 数据库各级模式的形成
  - 需求分析阶段：综合各个用户的应用需求（现实世界的需求）。
  - 概念设计阶段：概念模式（信息世界模型），用E-R图来描述。
  - 逻辑设计阶段：逻辑模式、外模式。
  - 物理设计阶段：内模式。