第11章　数据库设计

2012.05

目录 Contents

11.1 概述

11.1.1 信息社会与数据库

❖ 信息社会与企业信息化

- 信息成为社会的重要资源；
- 社会构架逐步从“金字塔式”向“网络化”过渡；
- 组织/企业动态化、边界模糊化，企业信息化：
  - 虚拟企业（Virtual Organization/Enterprise）：

    根据需要把优势互补的企业（合作伙伴）联合起来，以最有效、最经济的方式参与市场竞争、迅速响应瞬息万变的市场需求，提高竞争力。
  - 企业信息化：

    指企业如何与环境相协调，开发、利用好信息资源，使生产经营活动借助于信息的及时处理、顺畅流通而高质量、高效率的运作。
  - 企业信息化的根本任务：建立企业集成信息系统（EIIS）。

11.1 概述

❖ 数据中心原理

☙ 人们的认知过程：

重硬件、轻软件 → 重软件、轻数据 → 重数据

（60’ s-70’s） （70’s-80’s） （90’s-）

☙ 数据中心原理：

- James Martin 80年代提出；
- 企业集成信息系统应以数据为中心：

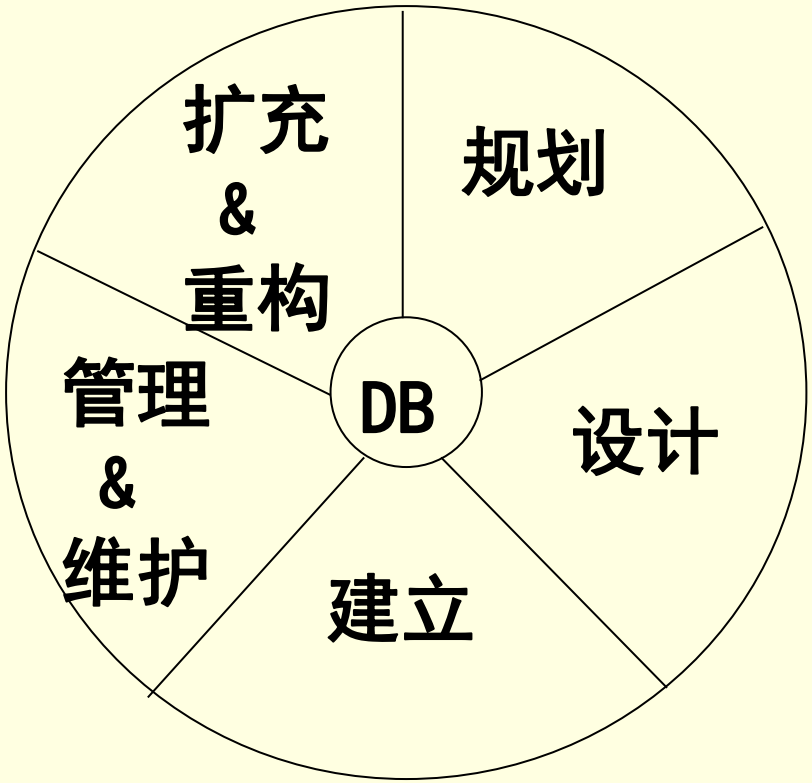
  数据类是相对稳定的；处理是多变的；集成化的数据环境。
- 数据规划、数据库设计&管理成为重点、难点；
- 西方企业中的重要角色：

  - CIO（Chief Information Officer）
  - DA（Data Administrator）
  - DBA

11.1 概述

11.1.2 数据库的生命周期

- 生命周期



11.1 概述

❖ 各阶段主要工作

☙ 规划（Planning）

- ➢应用范围确定，应用环境分析
- ➢投资估算，风险/效益分析
- ➢数据标准化工作（信息分类与编码）
- ➢DBMS及其支撑环境的规划、选择与配置
- ➢人员的培训与配备

☙ 设计（Design）

- ➢ 定义需求
- ➢ 数据建模
- ➢ 设计模式（外、逻辑、内）
- ➢ 设计完整性约束
- ➢ 设计典型应用

11.1 概述

- 建立（Build）
  - DBMS实现（数据模式、完整性约束、安全模式）
  - 加载数据
  - 应用培训
- 管理&维护（Administration & Maintenance）
  - 性能监视，DB调整
  - 日常管理与维护（安全、完整性、备份与恢复）
  - DB重组（Reorganization）
- 扩充&重构（Extending & Restructuring）
  - DB扩充
  - DB重构

11.1 概述

11.1.3 数据库设计

❖ 任务与目标

☙任务：

根据企业的需求（信息、处理需求）及数据库支撑环境（硬件、网络平台，OS平台，DBMS）的特点，设计出满足要求的数据模式（外模式、逻辑/概念模式、内模式）及典型应用。

☙目标：

对常用的、大多数应用能使用方便、性能满意。

11.1 概述

❖ 方法与特点

- 方法：

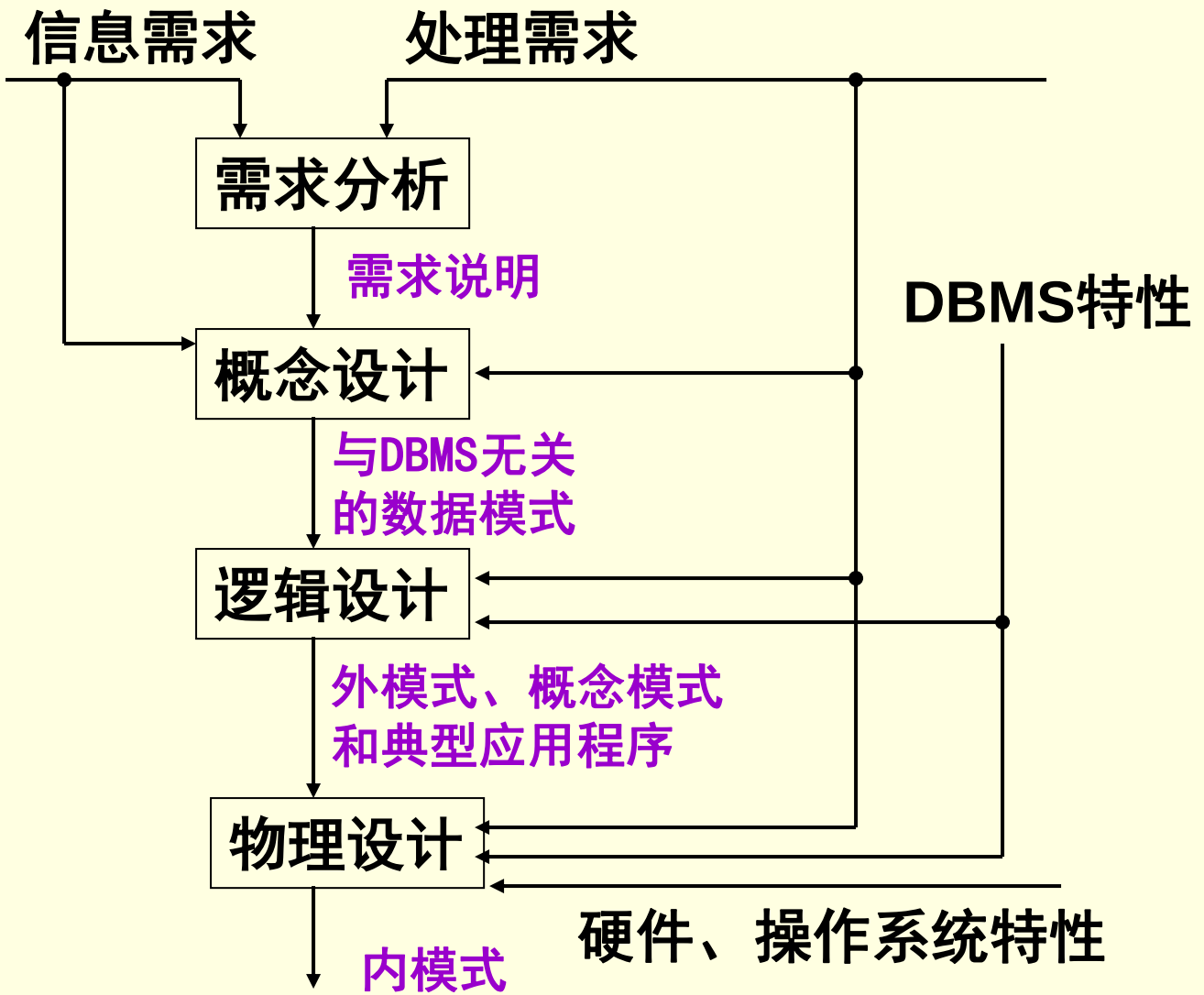
面向数据的方法-----信息工程方法（IEM）：

注重企业全局的数据规划和对企业业务目标的理解，在此基础上设计全局数据模式，作为企业集成信息系统的基础。

- 特点：反复性 / 试探性 / 分步进行

11.1 概述

过程与步骤


信息需求
处理需求
需求分析
需求说明
概念设计
与DBMS无关
的数据模式
逻辑设计
外模式、概念模式
和典型应用程序
物理设计
内模式
DBMS特性
硬件、操作系统特性

11.1 概述

- 需求分析（Requirements Analysis）
  - 定义需求规格说明
  - 构造元数据纲目库（Metadata Repository）
- 概念设计（Conceptual Design）
  - 数据建模（E/R）
- 逻辑设计（Logical Design）
  - 模式转换（E/R to Relational）
  - 模式规范化
  - 完整性约束设计
  - 外模式设计
  - 典型应用设计
- 物理设计（Physical Design）
  - 内模式设计

目录 Contents

11.2 需求分析

11.2.1 分析的过程

11.2 需求分析

11.2.2 提交的成果

- 需求规格说明:
  - 初步的实体/联系图（E/R）
  - 业务功能层次图（BFH）
  - 功能/实体、功能/业务单位、实体/业务单位矩阵
  - 数据流图（DFD）
  - 数据分布、约束、输出格式需求
  - 数据量，功能使用频率，用户期望性能
  - 各种策略：审计，控制，后备/恢复，转换，etc.

11.2 需求分析

- 元数据纲目库：

  对每个数据：

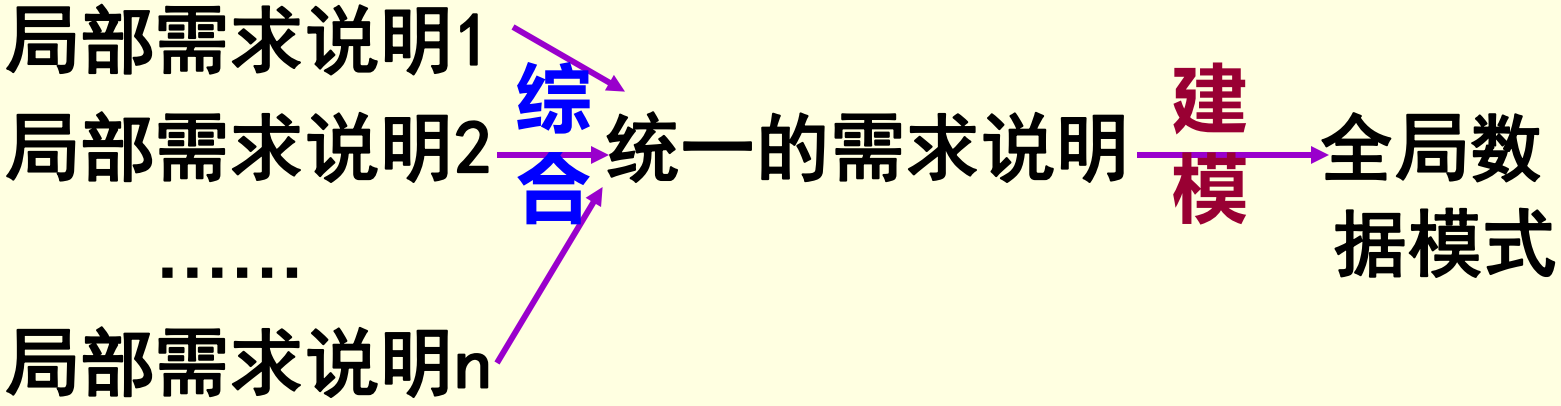
  - 数据名
  - 类型/长度
  - 编码规则
  - 更新要求
  - 使用频率
  - 数据量
  - 语义约束
  - 保密要求

目录 Contents

11.3 概念设计

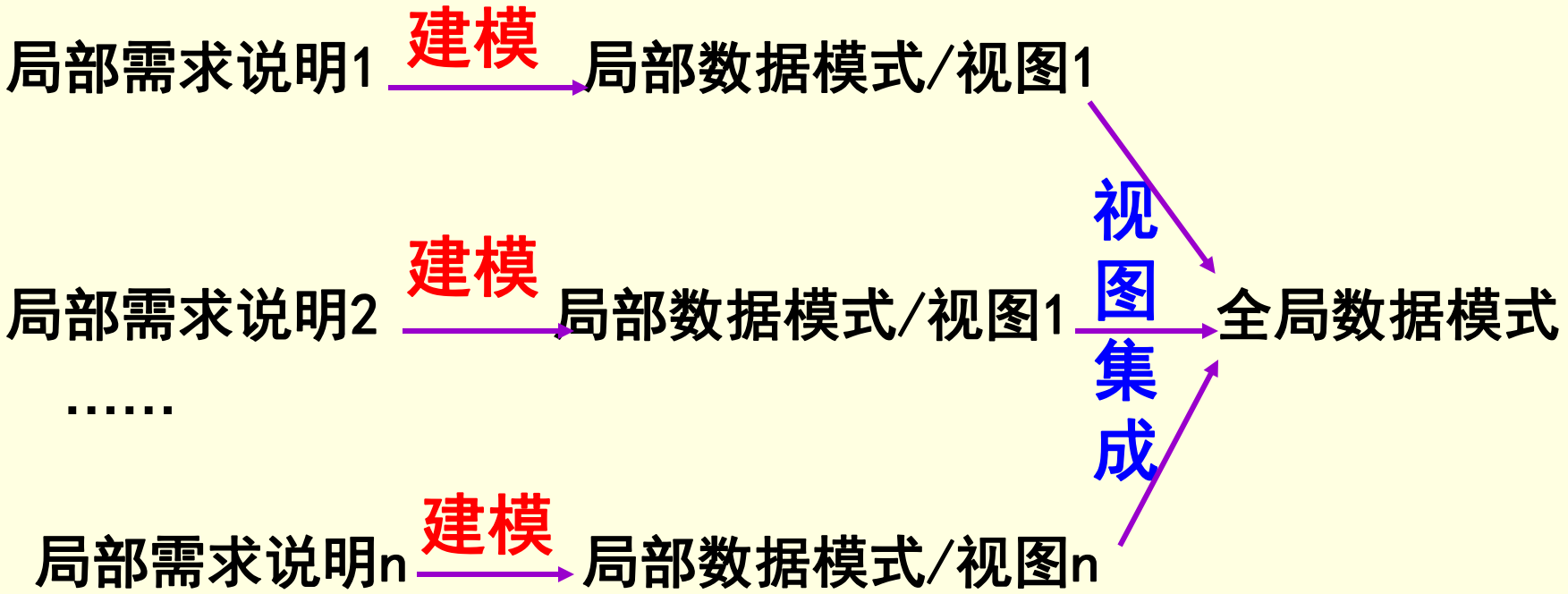
11.3.1 概述

- 任务　　数据建模 / 企业建模。
- 技术　　实体/联系建模（E/R Modeling）。
- 方法
  - 集中式模式设计法（Centralized Schema Design Approach）：



11.3 概念设计

- 视图集成法（View Integration Approach）：



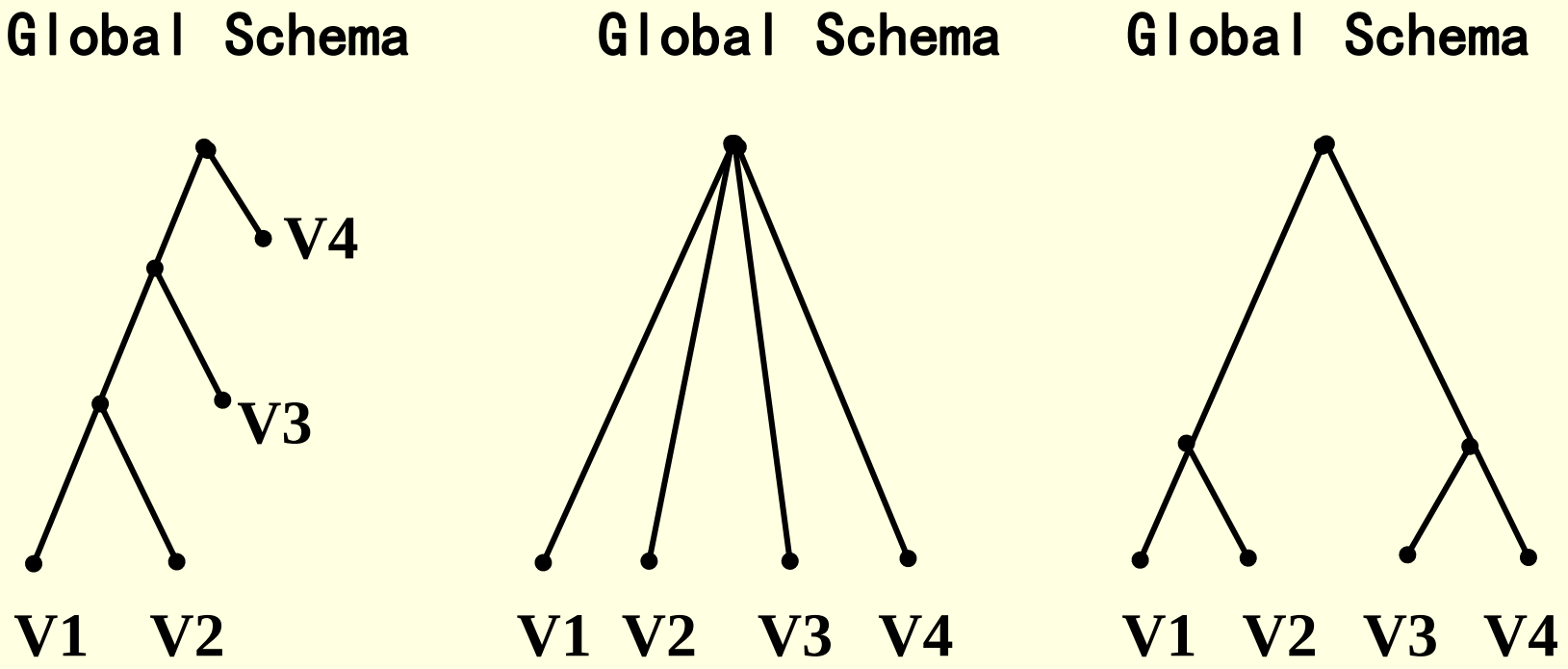
11.3 概念设计

11.3.2 视图集成

- **步骤**
  - **确认视图中的对应（“同一”）与冲突（“矛盾”）**
    - 命名冲突
    - 概念冲突
    - 域冲突
    - 约束冲突
  - **修改视图，解决部分冲突**
    - 修改视图，以尽可能地调和视图间的各种“冲突”
  - **合并视图，形成全局模式（Global Schema）**
    - 合并各视图中的对应部分、保留特殊部分、删除冗余部分，力求使全局模式简明清晰

11.3 概念设计

- 视图合并集成的策略



梯形集成　　n元集成　　平衡集成

目录 Contents

11.4 逻辑设计

11.4.1 概述

- **任务**

  设计数据库的**逻辑（概念）模式和外模式**。

- **技术**

  - E/R数据模式向关系数据模式转换；
  - 关系模式规范化（已学过）；
  - 模式调整；
  - 外模式设计。

11.4 逻辑设计

11.4.2 E/R向关系转换

- 基本规则：

实体集 / 属性 —转换→ 关系模式 / 属性

联系集 / 属性 —转换→ 置于参与联系的某个实体的关系模式中 / 属性

或：单独的关系模式/ 属性

11.4 逻辑设计

- 实体的转换：
  - 模式及属性的命名：易读、好记。

    e.g. 合同的关系模式：

    contract（code, name, first-part, second-part,sign-date, total-price, ...）

    ht（bm, mc, jf, yf, qdrq, zj, ...）

    -------不好！
  - 属性域的处理：选择最合适的数据类型/自定义数据类型。

11.4 逻辑设计

- **非原子属性的处理：**

  （对集合属性）纵向展开；

  （对元组属性）横向展开。

- **键的处理：**

  实体集的键成为关系模式的键（可选定其中之一作主键PK）。

- **弱实体的处理：**

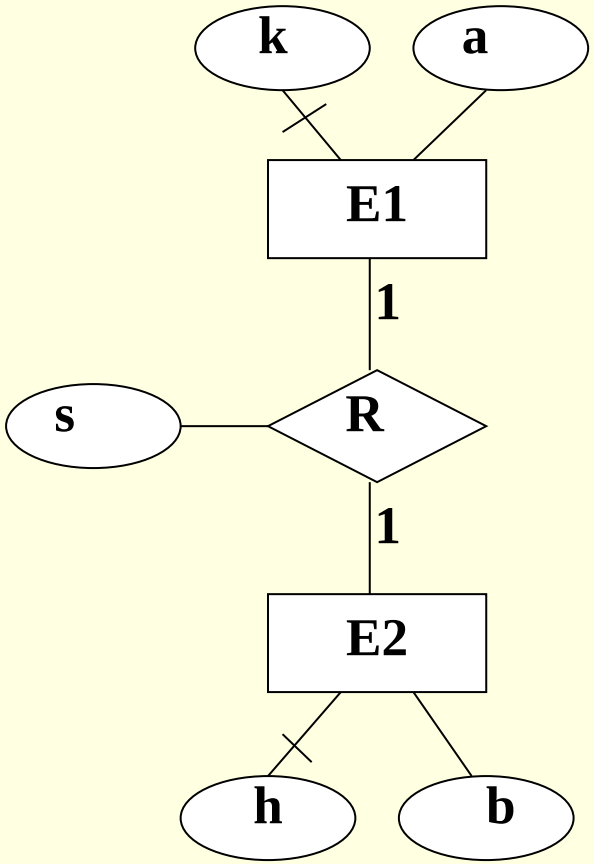
  相应的关系模式中应包含其所有者实体的（主）键。

e. g. 家属（职工号，姓名，年龄，与职工的关系）

Key = {职工号，姓名}；FK = {职工号}。

11.4 逻辑设计

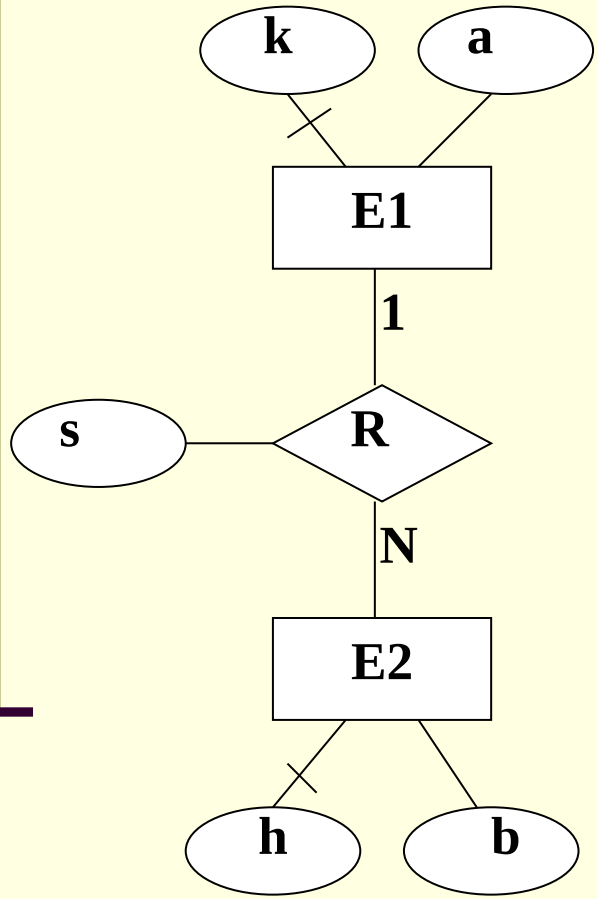
- 联系的转换：
  - 1:1联系：



若E1全参与：	若E1不是全参与：
R1（k，a，h，s）	R1（k，a）
	R3（k，h，s）
R2（h，b）	R2（h，b）
注：——PK ～～FK ····· 候补键	

11.4 逻辑设计

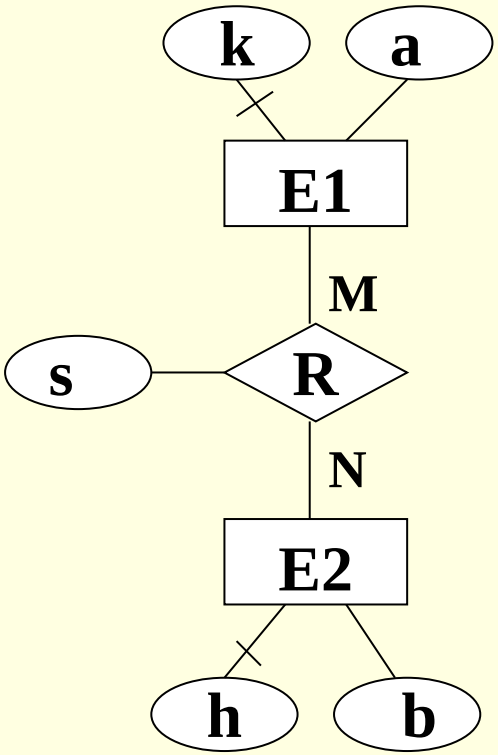
1:N联系：



若E2全参与：	若E2不是全参与：
R1（k，a）	R1（k，a）
	R3（h，k，s）
R2（h，b，k，s）	R2（h，b）
注：— PK ～～FK	

11.4 逻辑设计

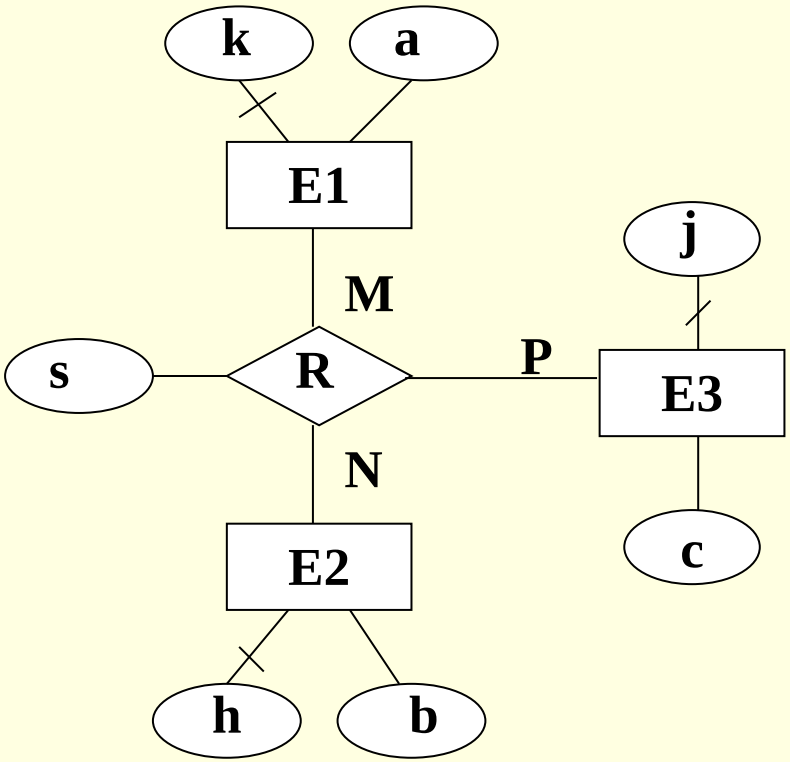
M:N联系：



不管是否全参与：
R1（k，a）
R3（h，k，s）
R2（h，b）

11.4 逻辑设计

- M:N:P联系：



不管是否全参与：
R1（k，a）
R2（h，b）
R3（ j，c）
R4（k，h，j，s ）

11.4 逻辑设计

11.4.3 关系模式的规范化（已学过）

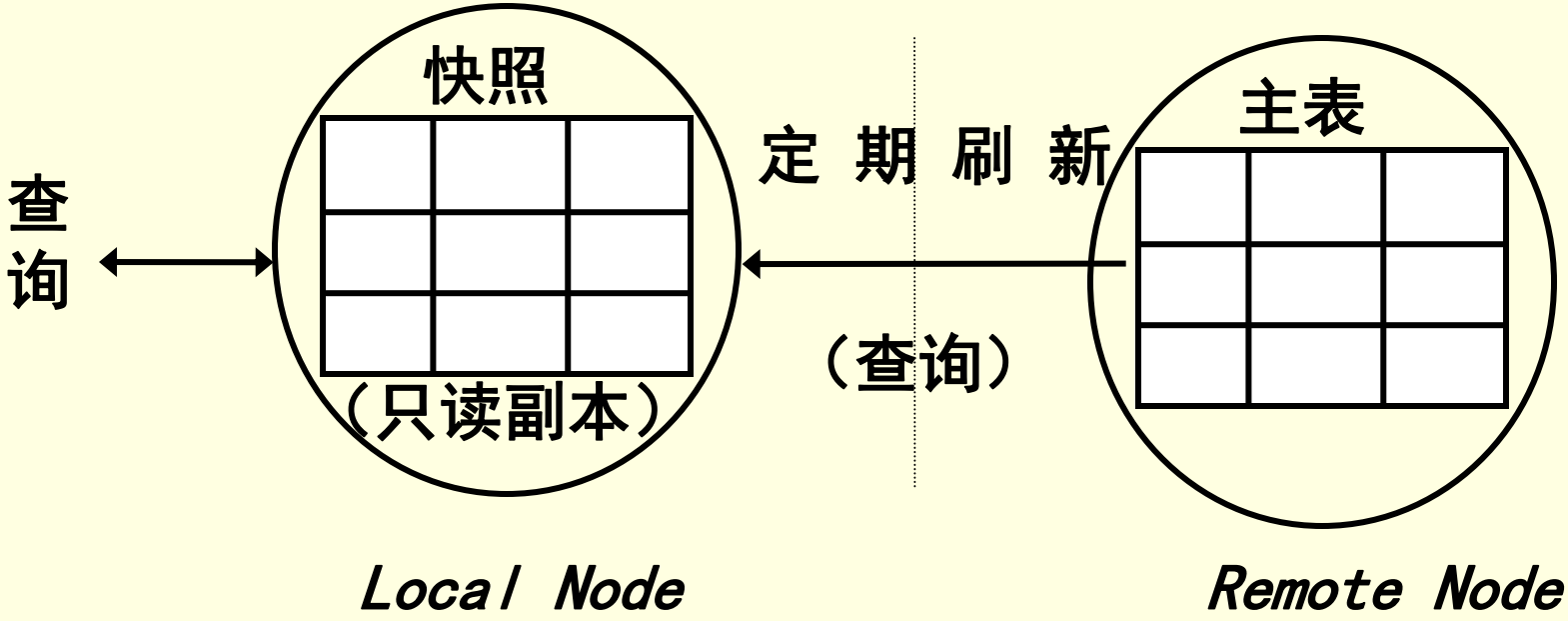
11.4.4 模式的调整

- 为改善DB性能的调整：
  - 减少连接运算：

    逆规范化（Denormalization）。
  - 减小表的大小：

    分割关系（水平分割；垂直分割）。
  - 尽可能使用快照（Snapshot）：

11.4 逻辑设计



11.4 逻辑设计

- 为节省存储空间的调整：
  - 减少属性的长度：

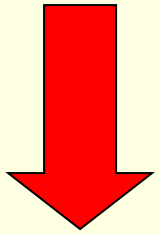
    用编码代替属性值；用缩写代替全称。

  - 用替身（Dummy）属性减少重复数据所占的存储空间：

11.4 逻辑设计

e.g.

学生	家庭人均收入	有无其他经济来源
张三	0 ~ 1000元/人.月	无
张四	0 ~ 1000元/人.月	无
张五	0 ~ 1000元/人.月	无
张六	0 ~ 1000元/人.月	有
李三	1000 ~ 2000元/人.月	无
王五	2000元/人.月以上	有



11.4 逻辑设计

等级	家庭人均收入	有无其他经济来源
1A	0 ~ 1000元/人.月	无
1B	0 ~ 1000元/人.月	有
2A	1000 ~ 2000元/人.月	无
2B	1000 ~ 2000元/人.月	有
3B	2000元/人.月以上	无
3B	2000元/人.月以上	有

学生	家庭经济状况等级
张三	1A
张四	1A
张五	1A
张六	1B
李三	2A
王五	3B

11.4 逻辑设计

11.4.5 外模式的设计

- 依据：

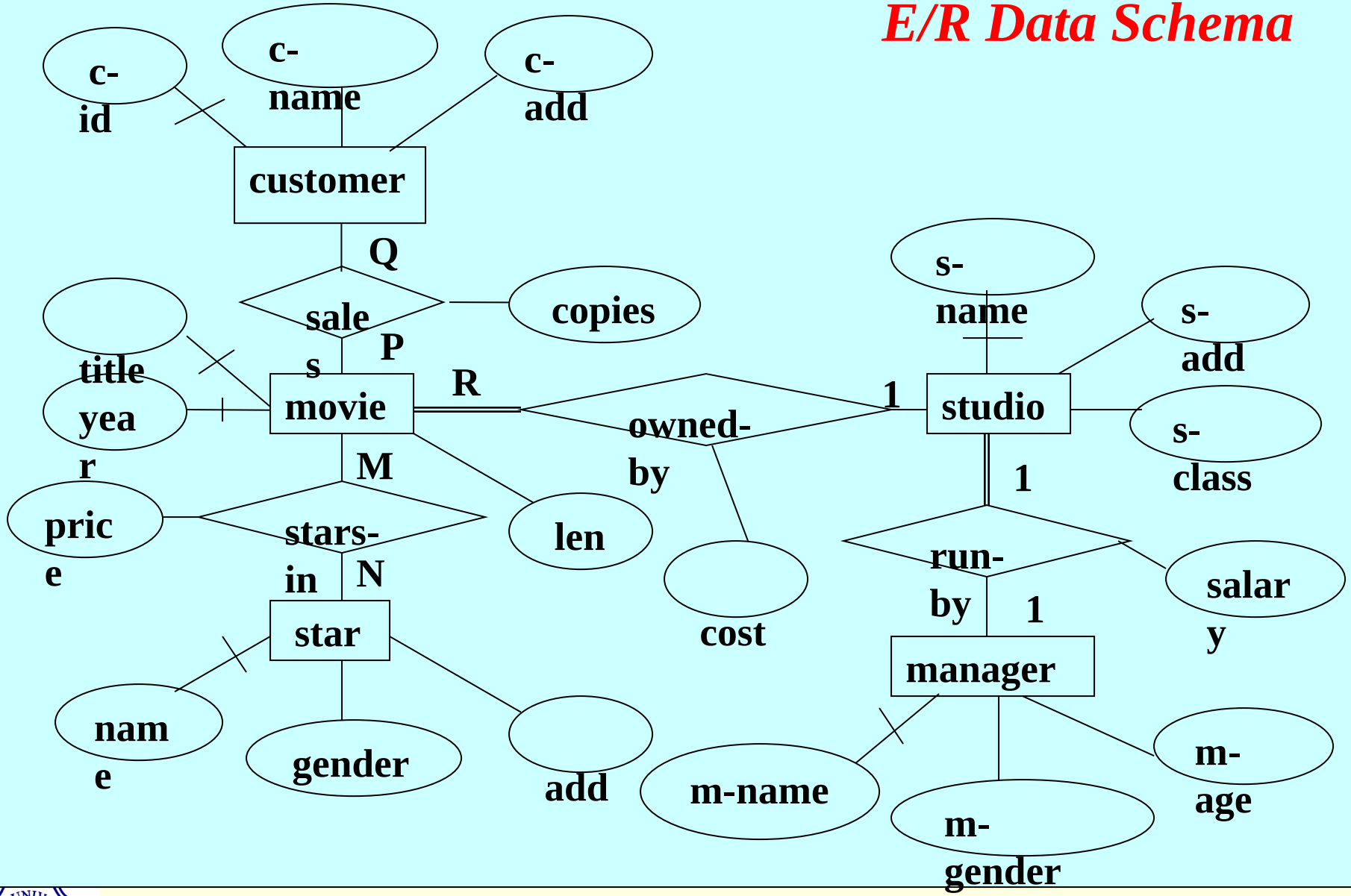
  局部E/R数据模式（即用户视图）。

- SQL实现：

  基表 + 视图。

[例] E/R to Relational

E/R Data Schema



11.4 逻辑设计

Relational Schemas: **(注： — PK ∿∿ FK)**

movie (title, year, len, s-name, cost)

star (name, gender, add)

stars-in (title, year, name, price)

customer (c-id, c-name, c-add)

sales (title, year, c-id, copies)

studio (s-name, s-add, s-class, m-name, salary)

manager (m-name, m-gender, m-age)

目录 Contents

11.5 物理设计

11.5.1 概述

❖ 任务

设计数据库的内模式（即存储模式）。

目的是：（1）提高DB的性能；（乃主要目的！）

（2）有效地利用存储空间。

❖ 技术

☙存储机制的选择：

有四种选择：无簇表：(1) 表

(2) 索引的表

簇　表：(3) 散列簇表

(4) 索引簇表

☙分区设计：数据在多个磁盘上合理分布，以合理地安排I/O，从而提高DB性能。

11.5 物理设计

11.5.2 存储机制的选择

- **索引的选择：**
  - **建立：**

    一般，PK及UNIQUE属性列上的索引是DBMS自动建立的；

    其他属性列上的索引需用户（一般是DBA）手工建立。
  - **使用**：索引的使用由DBMS自动决定，即对用户是透明的。

11.5 物理设计

❶下列情况，宜在有关属性列上建索引：

——即用(2)索引的表

- 常用来连接（Join）相关表的列上宜建索引

  常用的连接是通过PK/FK来实现的，PK列上DBMS自动建立了索引，因此，最好在FK列上也建索引（假如对FK列的更新不太频繁的话）

- 以读为主的表，其常用查询涉及的列上宜建索引。

  前提：范围查询满足条件的行数≤20%表行，等值查询满足条件的行数≤５%表行；且条件不是IS NOT NULL

- 对“只需访问索引块而不需访问数据块的查询”是常用查询的表，其相应列上宜建索引

  (1)查询某属性值的MIN, MAX, AVG, SUM, COUNT等组函数值，且无GROUP BY子句

  (2)查询某属性值是否 (NOT) EXISTS

11.5 物理设计

❷下列情况，不宜在有关属性列上建索引：

- 很少出现在查询条件中的列。（没意义）
- 属性值很少的列。（没必要）
- 太小的表。（没必要）
- 属性值分布严重不均匀的列。（反而不如全表扫描）
- 经常频繁更新的表 / 列。（系统维护索引的开销太大）
- 过长的列。（难实现。如：Oracle规定不能在LONG或LONG RAW类型的列上建索引）

11.5 物理设计

- **簇集的选择：**
  - 建立：

    簇集的建立需用户（一般是DBA）手工完成。
  - 使用：

    簇集的使用由DBMS自动决定，即对用户是透明的。（实际中常用散列簇集，较少用索引簇集）

11.5 物理设计

❶下列情况，宜在有关属性列上建簇集：

- 更新较少的表，且对其只进行等值查询时，宜建“散列簇集”。

  ——即用(3)散列簇表

- 更新较少的表，且对其既进行等值查询又进行范围查询时，可建“索引簇集”。

  ——即用(4)索引簇表

❷下列情况，不宜在有关属性列上建簇集：

- 频繁更新的表，则宁可用“普通的表”。
- 更新较少的表，但对其只进行范围查询时，则宁可用“索引的表”。

11.5 物理设计

11.5.3 分区设计的原则

- 水平分割关系，以减少访盘冲突、提高I/O并行性
- 分散热点数据（Hot Spot Data），均衡I/O负荷
- 保证关键数据（e.g., DD）的快速访问，缓解系统瓶颈。

＊ 实际系统中，如何简单地实现分区设计？

一个DB可被划分为一个或多个称表空间（Table Space）的逻辑存储单元，一个表空间所对应的一个或多个数据文件（Data File）可物理存储于不同的磁盘上。