

数据库系统原理

教程：数据库系统概论（第5版）

结合：CMU 15-445/645 INTRO TO DATABASE SYSTEMS

华中科技大学 计算机学院

左琼



第七章 数据库设计

Principles of Database Systems

第七章 数据库设计

7.1 数据库设计概述

7.2 需求分析

7.3 概念结构设计

7.4 逻辑结构设计

7.5 数据库的物理设计

7.6 数据库实施和维护

7.7 小结

概念模型设计的方法和步骤

1. 概念结构设计的方法

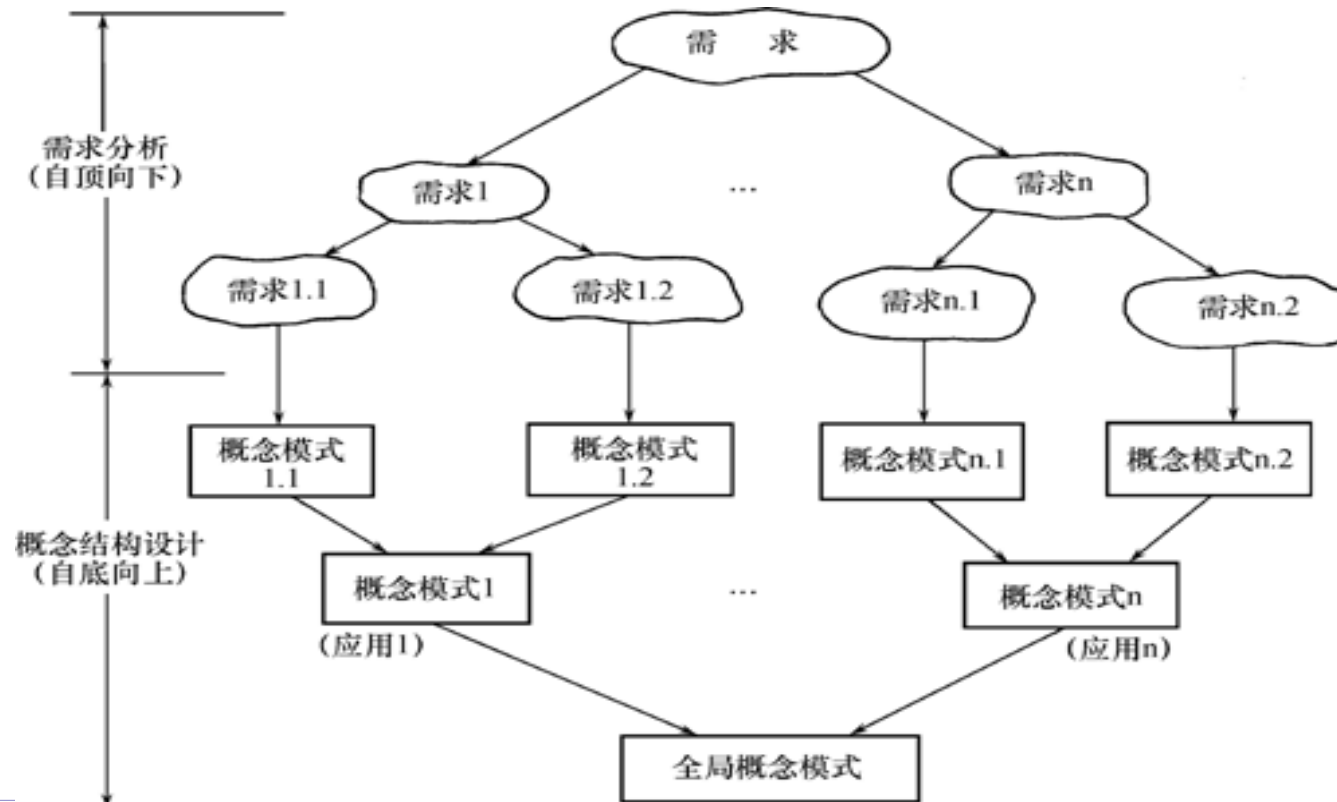
设计概念结构的E-R模型可采用四种方法。

- (1) **自顶向下** 先定义全局框架，再逐步细化。
- (2) **自底向上** 先定义局部E-R模型，然后将它们集成，得到全局E-R模型。
- (3) **逐步扩张** 先定义核心概念E-R模型，然后以滚雪球的方式逐步生成其他概念结构E-R模型。
- (4) **混合策略** 自顶向下和自底向上相结合的方法。

概念模型设计的方法和步骤

□ 常用策略:

- 自顶向下地进行需求分析
- 自底向上地设计概念结构

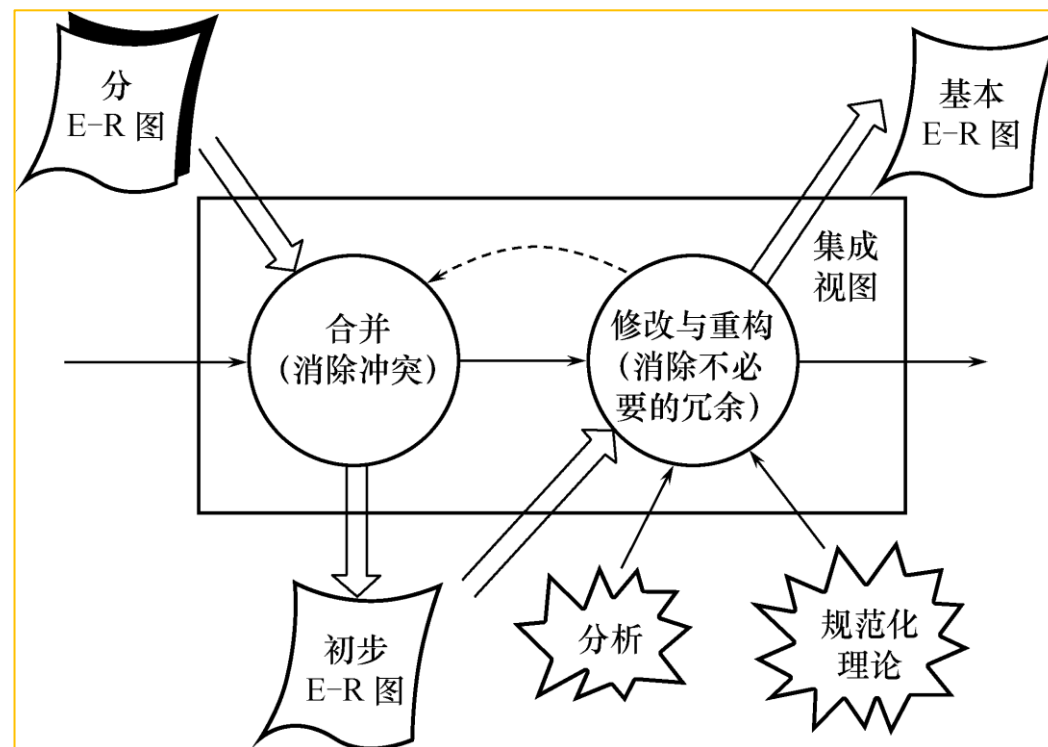
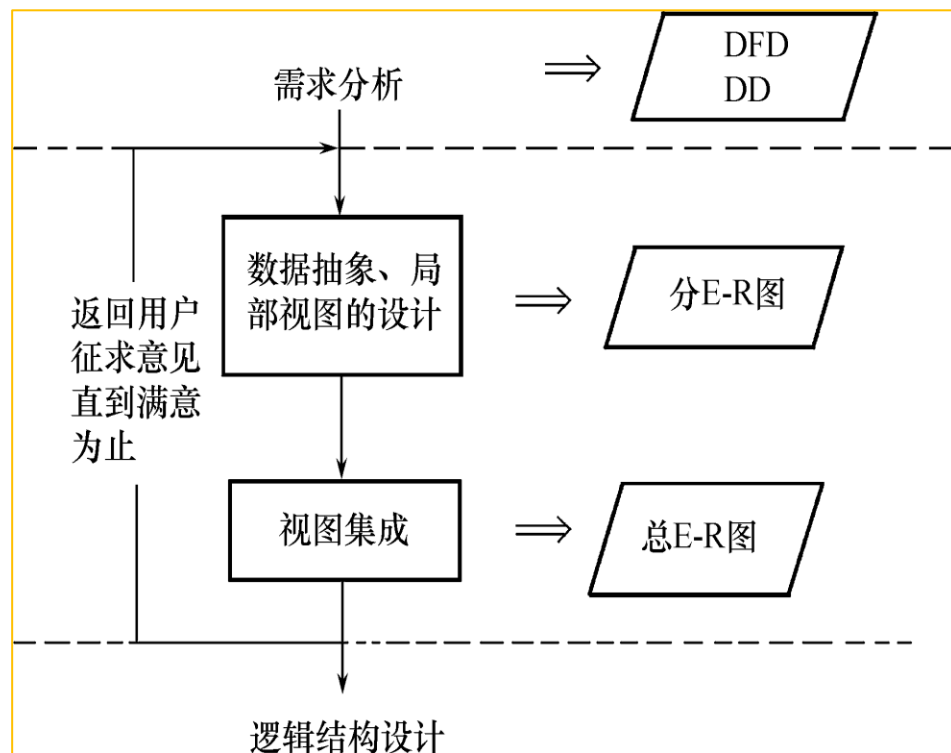


概念模型设计的方法和步骤

□ 自底向上设计概念结构的步骤

第1步：抽象数据并设计局部视图

第2步：集成局部视图，得到全局概念结构



步骤1:数据抽象和局部设计

□ 概念设计的第一步：利用抽象机制对需求分析阶段收集的数据进行分类、组织形成实体、实体的属性，标识实体的码，确定实体之间的联系类型，设计分E-R图。

□ 设计分E-R图：

选择局部应用，确定范围原则：

- 1) 相对独立；
- 2) 内部联系较紧密；
- 3) 与外部联系相对较少。

逐一设计分E-R图：利用抽象机制形成实体、属性，确定联系等。

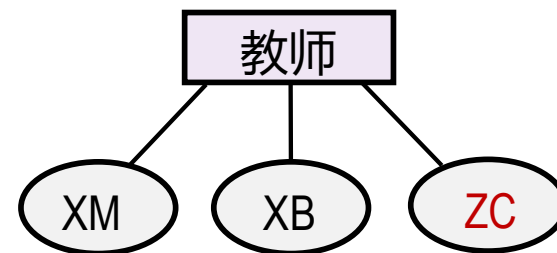
划分实体和属性的准则

□ 划分遵循**两条原则**：

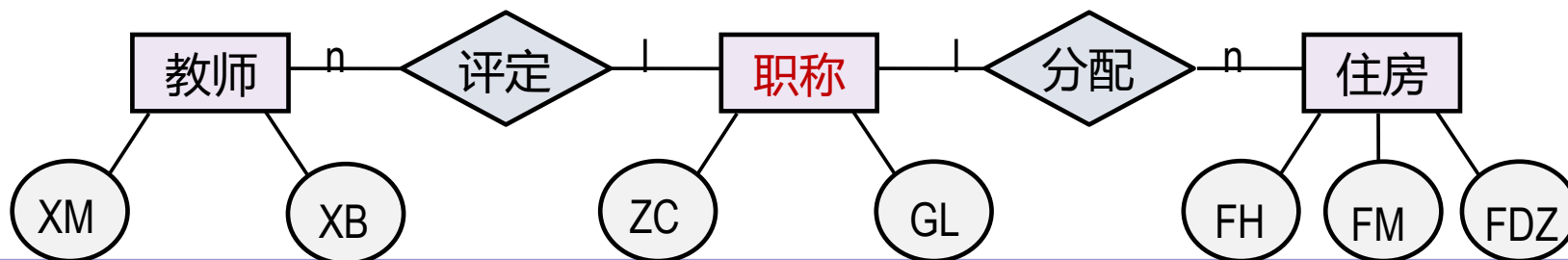
- ① 实体具有描述信息，而属性没有。属性必须是不可分的，不能包含其他属性。可再分者为实体。
- ② 联系只能发生在实体之间。

一般情况下，凡能作为属性对待的，应尽量作为属性，以简化E-R图的处理。

【例】一般“ZC（职称）”做属性：

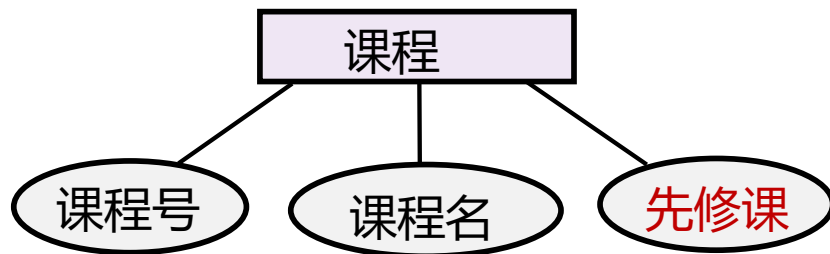


若职称与住房有联系则作实体：

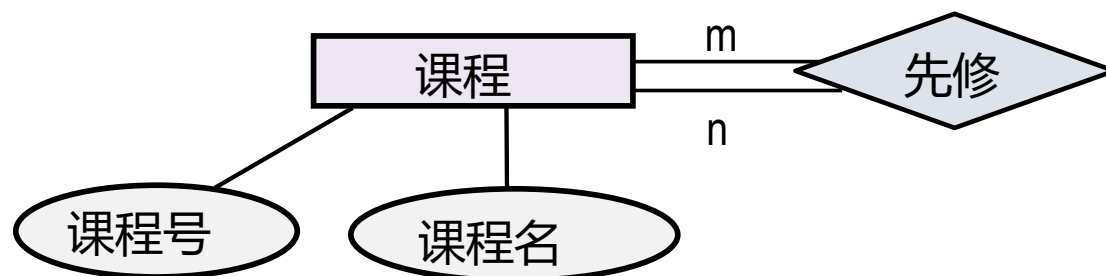


划分实体和属性的准则

□ [例] 一门课只存在一门直接先修课



若：一门课存在多个先修课



步骤2：集成局部设计，生成全局概念结构

□ 集成的方式：

- ① **多元集成法**：一次性将多个局部E-R图合并为一个全局E-R图（复杂）。
- ② **二元集成法**：首先集成两个重要的局部视图，以后用累加的方法逐步将一个新的视图集成进来。

□ 无论使用哪一种方法，集成均分成两个步骤：

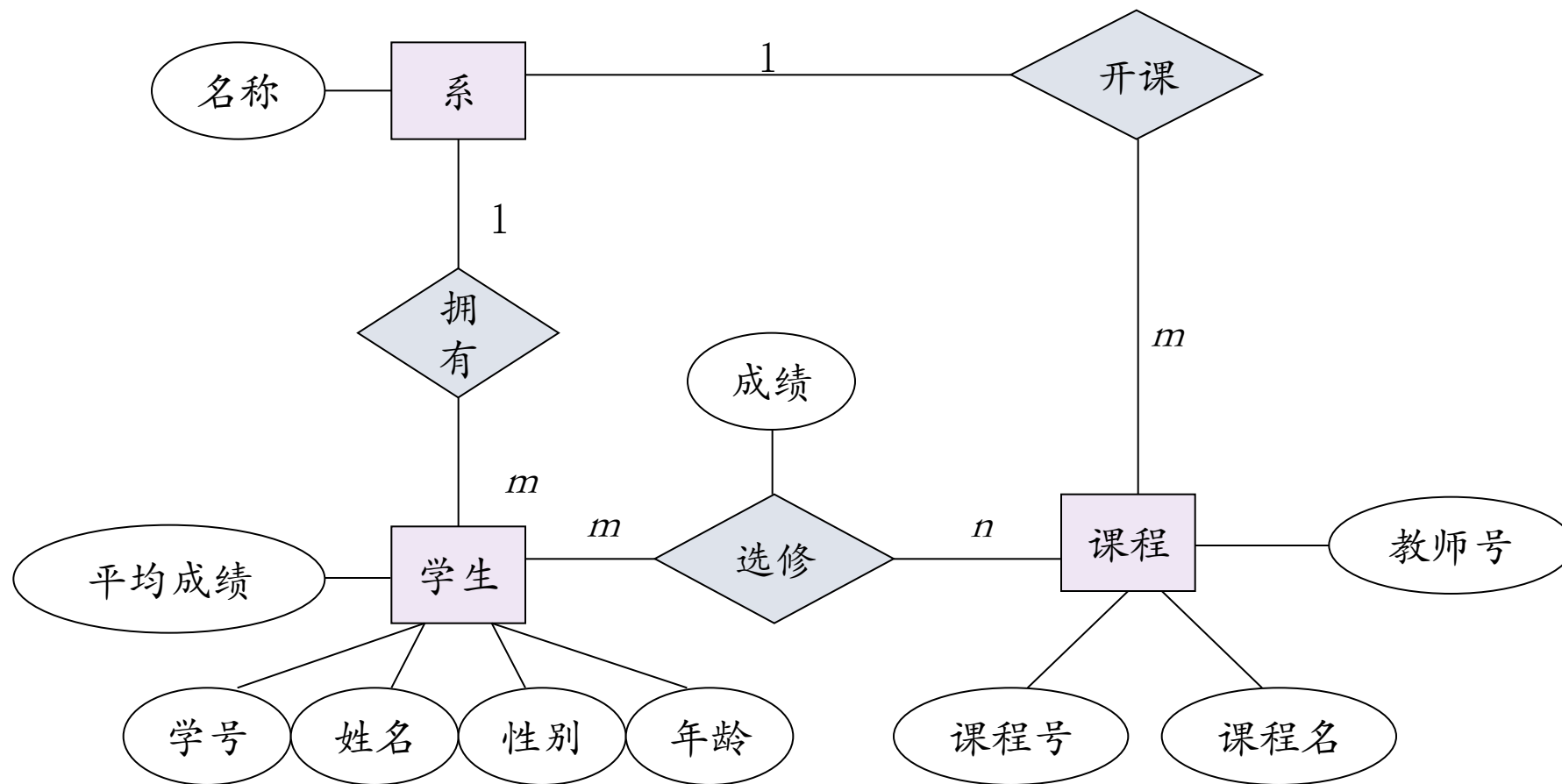
- ① **合并**：消除局部E-R图之间的**冲突**，生成**初步E-R图**。
- ② **优化**：消除**不必要的冗余**，生成**基本E-R图**。

局部E-R模型设计

- 举例：局部E-R模型设计。
- 在一个简单的教务管理系统中，有如下语义约束：
 - ①一个学生可选修多门课程，一门课程可为多个学生选修，因此学生和课程是多对多的联系；
 - ②一个教师可讲授多门课程，一门课程可为多个教师讲授，因此教师 and 课程也是多对多的联系；
 - ③一个系可有多个教师，一个教师只能属于一个系，因此系和教师是一对多的联系，同样系和学生也是一对多的联系。

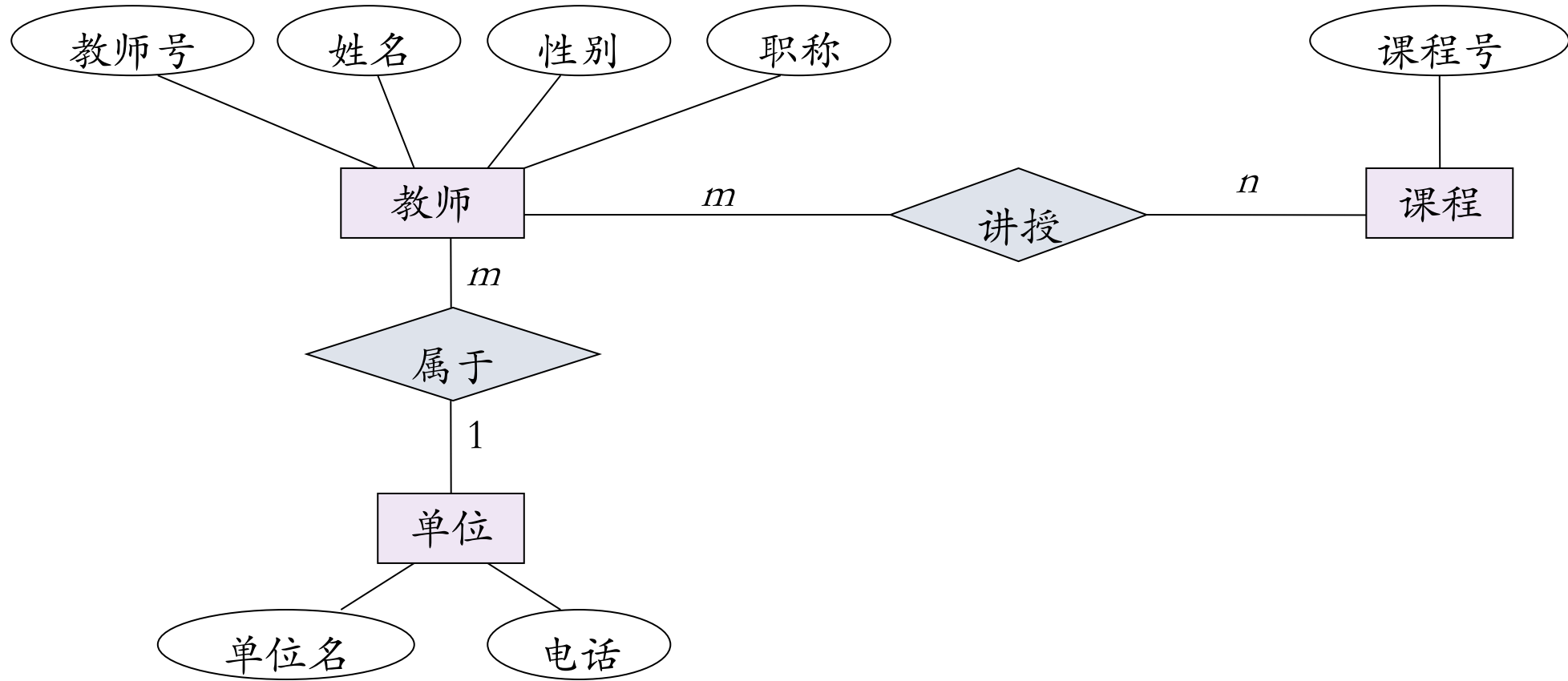
学生选课局部E-R图

□ 学生选课:



教师任课局部E-R图

□ 教师任课:



全局ER模型设计

□ E-R图中的冲突有三种：属性冲突、命名冲突和结构冲突。

① 属性冲突

■ **属性值域冲突**——即属性值的类型、取值范围或取值集合不同。

如：学号，数值型 或 字符型。

年龄，出生年月 或 整数。

■ **属性的取值单位冲突**

如：零件的重量，以：公斤为单位 或 斤为单位 或 克为单位。

属性冲突属于用户业务上的约定，必须与用户协商后解决。

全局ER模型设计

② 命名冲突

命名不一致可能发生在**实体名、属性名或联系名之间**，其中属性的命名冲突更为常见。

- **同名异义**——即同一名字的对象在不同的部门中具有不同的意义。

例如：“单位”在某些部门表示为人员所在的部门，而在某些部门可能表示物品的重量、长度等属性。

- **异名同义**——即同一意义的对象在不同的部门中具有不同的名称。

例如：对于“科研项目”，在财务部门被称为“项目”，而在科研处被称为“课题”。

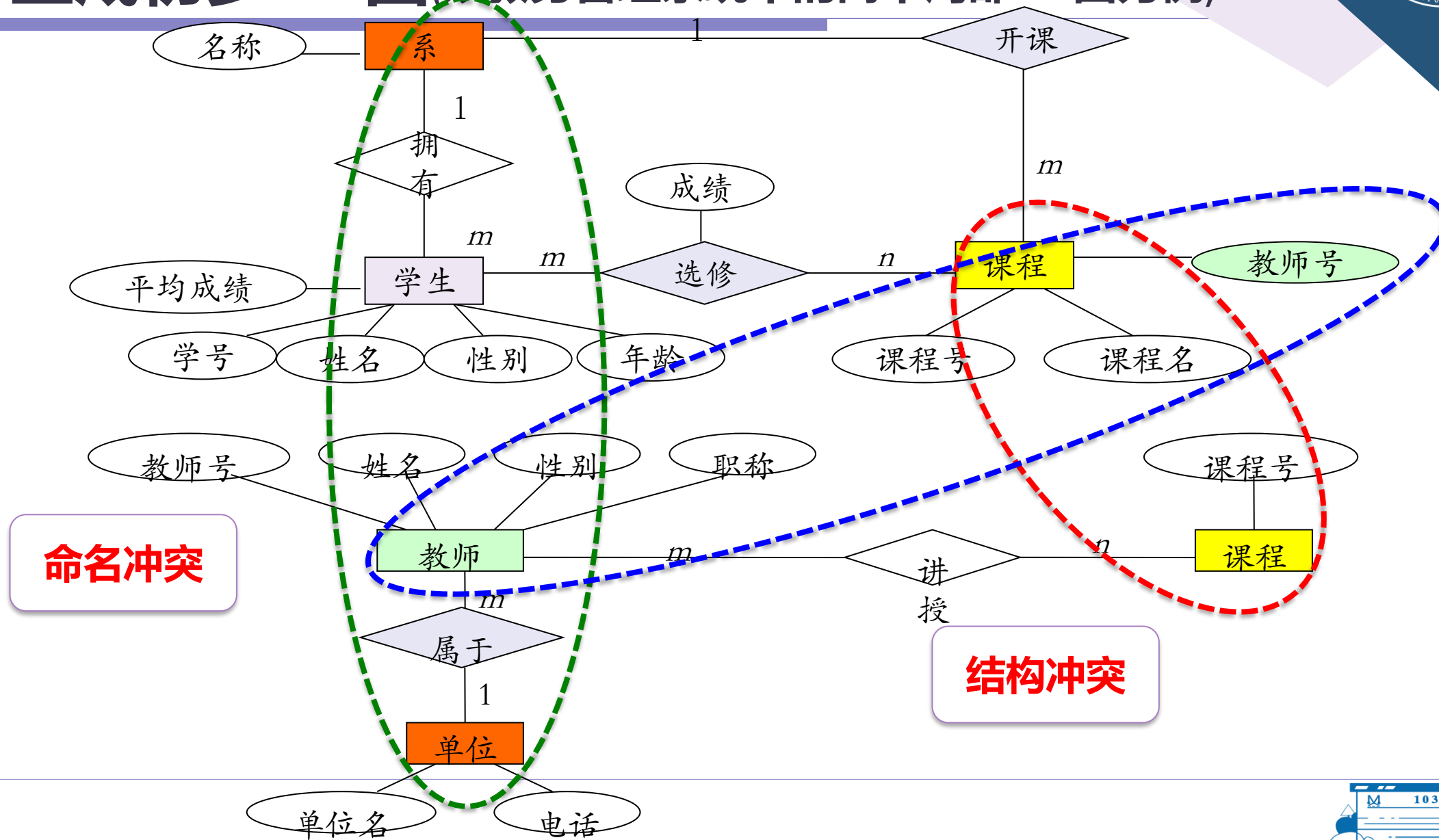
命名冲突的解决方法同属性冲突，需要与各部门协商、讨论后加以解决。

全局ER模型设计

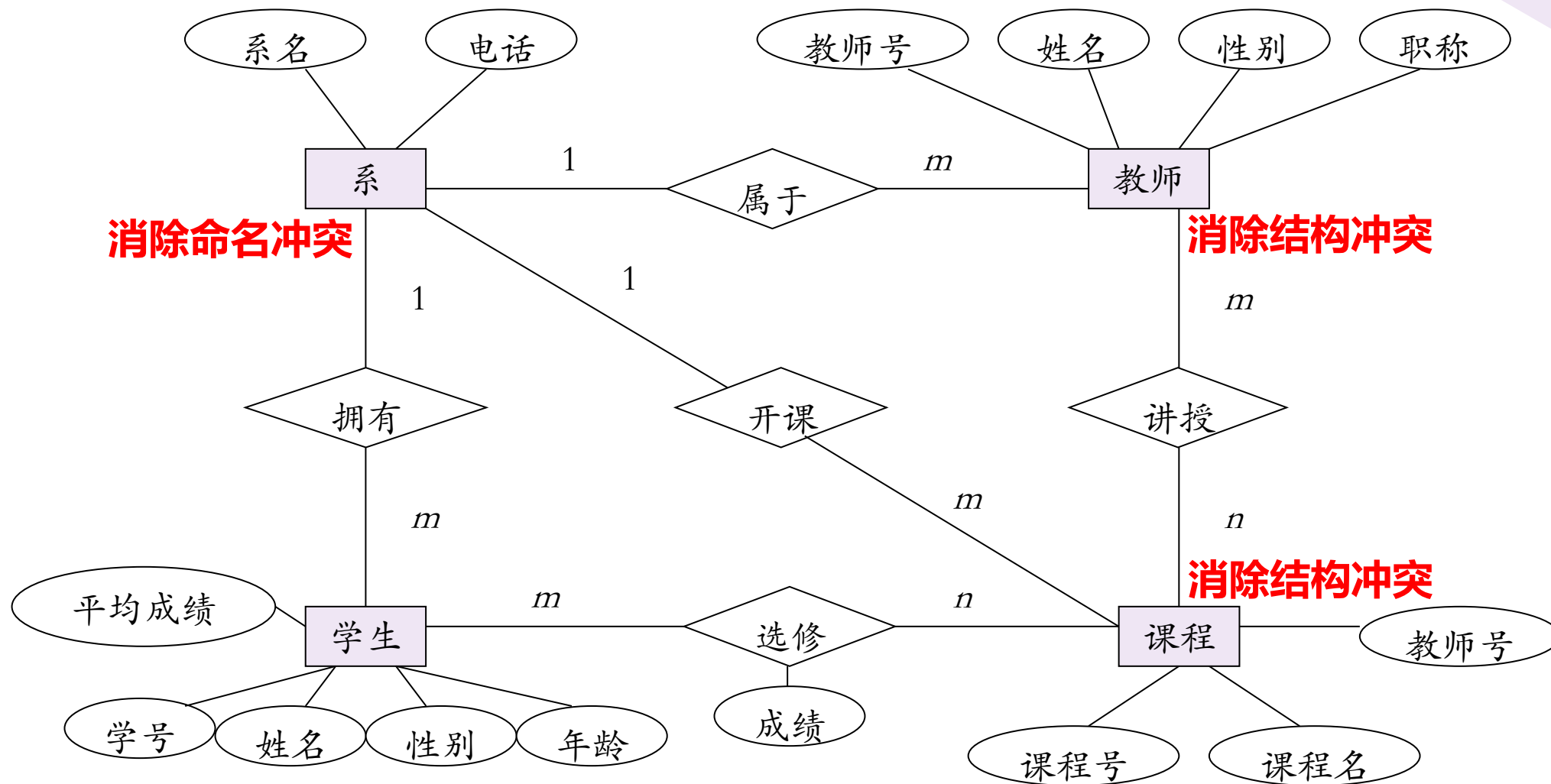
③ 结构冲突

- 同一对象在不同应用中有不同的抽象，可能为实体，也可能为属性。如，教师的职称在某一局部应用中被当作实体，而在另一局部应用中被当作属性。
- 解决方法：统一抽象级别（要符合7.3.3中所介绍的准则）
- 同一实体在不同应用中属性组成不同，可能是属性个数或属性次序不同。
解决办法：取各局部E-R图中的同名实体属性的并集，然后再适当调整属性的次序。
- 同一联系在不同应用中呈现不同的类型。如E1与E2在某一应用中是一对一联系，而在另一应用中是一对多或多对多联系，或在E1、E2、E3三者之间有联系。
解决办法：根据应用语义对实体联系的类型进行综合调整。

1. 生成初步E-R图(以教务管理系统中的两个局部E-R图为例)



初步E-R图

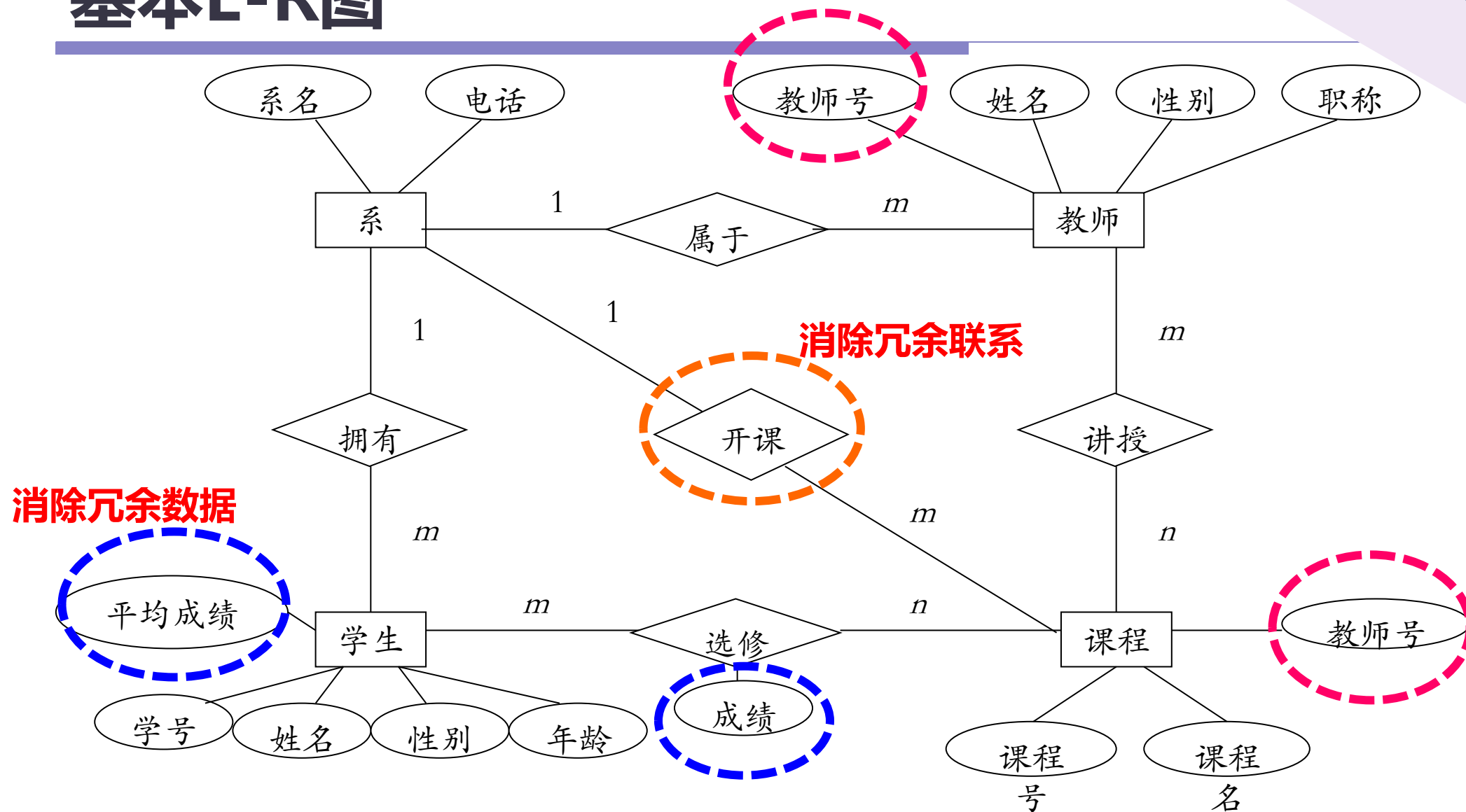


全局E-R图设计

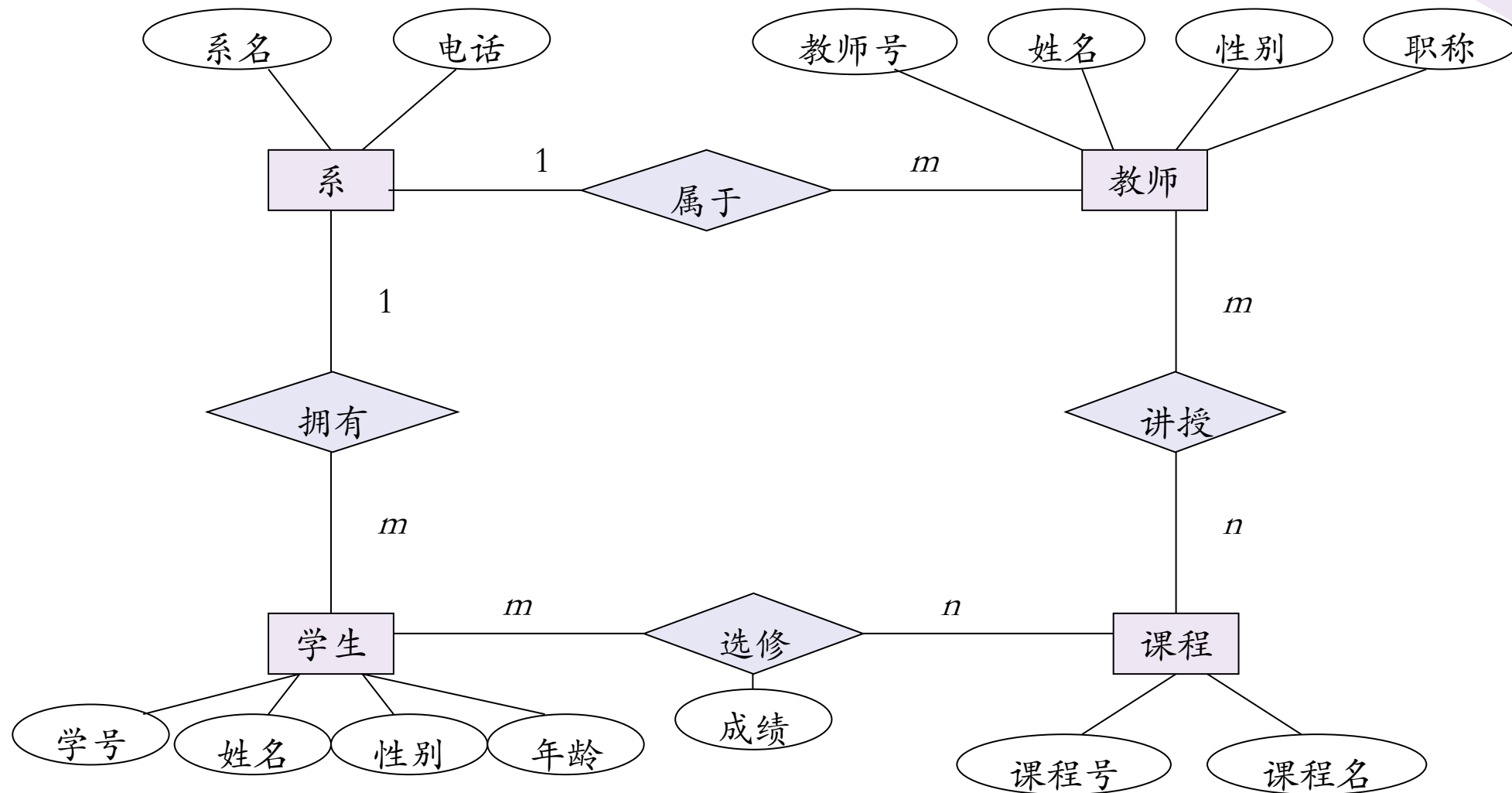
(2) 消除不必要的冗余，生成基本E-R图

- 冗余包括冗余的数据和冗余的联系。
 - 冗余的数据是指可由基本的数据导出的数据。
 - 冗余的联系是由其他的联系导出的联系。
- 消除了冗余的初步E-R图称为基本E-R图。
- 通常采用分析的方法消除冗余。数据字典是分析冗余数据的依据，还可以通过数据流图分析出冗余的联系。

基本E-R图



最终全局E-R图



概念结构设计小结

回顾

自底向上的设计方法:

1. 抽象数据并设计局部视图

1) 选择局部应用

2) 逐一设计分E-R图: 划分实体 / 属性, 关系类型的确定

2. 集成局部视图

1) 合并分E-R图, 生成初步E-R图

消除冲突: 属性冲突、命名冲突、结构冲突

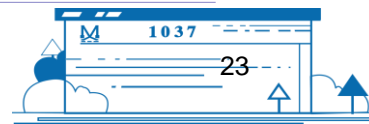
2) 修改与重构, 消除不必要的冗余, 设计生成基本E-R图

分析方法、规范化理论

3. 验证整体概念结构

案例：图书馆管理系统

- 请给出如下场景（满足1~4）的图书管理数据库的ER图设计：
 - 用户类型：读者、管理员
 - 管理员负责图书的管理和读者的管理；
 - 一个读者可以借多本书；一本书可以借给多个读者（馆藏多本）；
 - 一本书最迟1个月要归还；如果逾期还书，需按逾期天数罚款。
- 题目升级（如果要满足5~8，ER图如何设计？）：
 - 读者要有所属单位信息；
 - 图书管分为若干室，书按图书类型分别放在不同室的不同书架上；
 - 若第4点改为：读者可分为：学生、老师；学生最多可借有图书5本，教师最多可以借阅图书10本；
 - 借期30天，可续借1次，续借15天；逾期每册书每天罚款0.2元。



案例：图书馆管理系统(1)

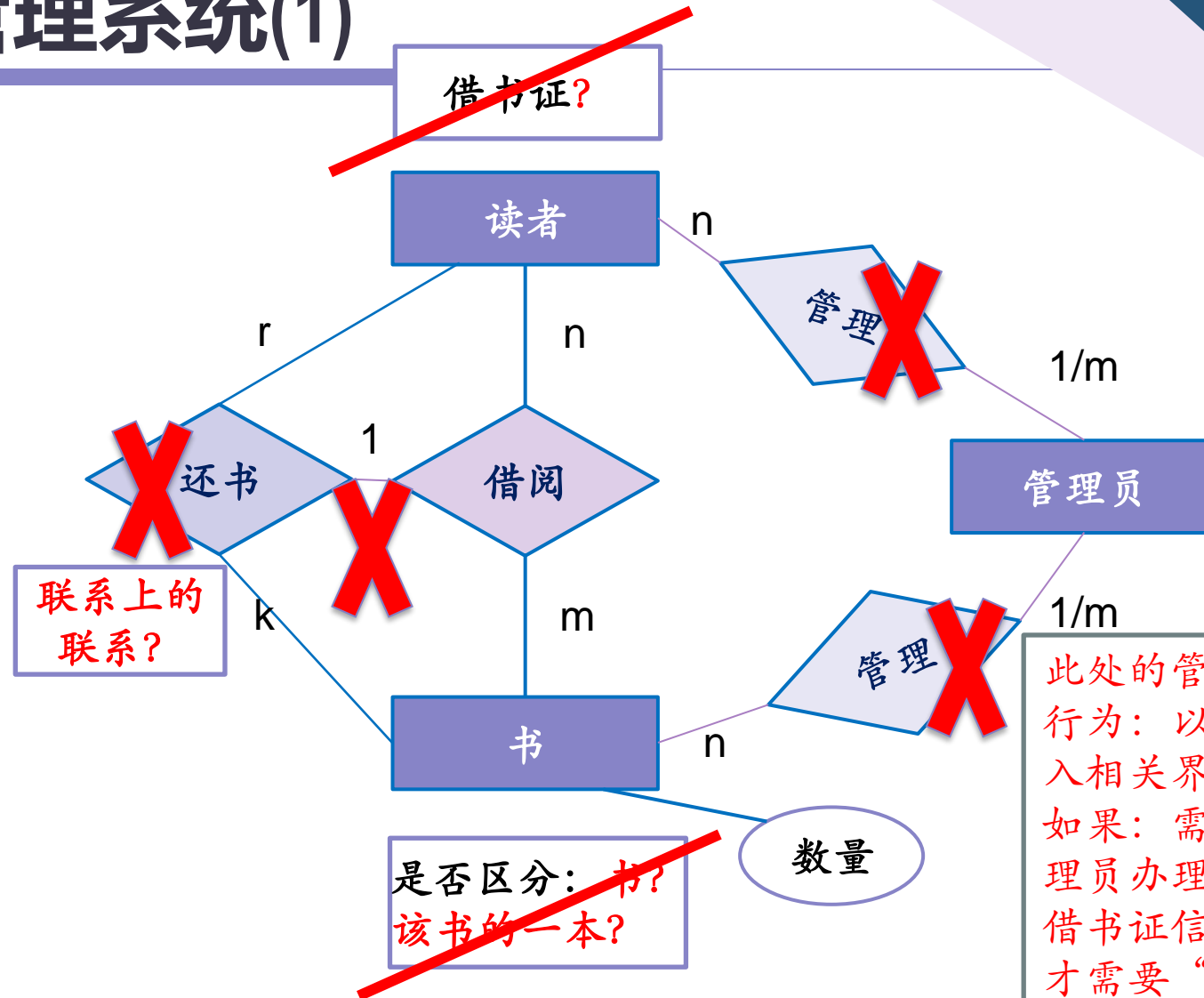
A. 请给出如下场景（满足1~4）的图书管理数据库的ER图设计：

1. 用户类型：读者、管理员

2. 管理员负责图书的管理和读者的管理；

3. 一个读者可以借多本书；一本书可以借给多个读者（馆藏多本）；

4. 一本书最迟1个月要归还；如果逾期还书，需按逾期天数罚款。



此处的管理是一个操作行为：以管理员身份进入相关界面操作。
如果：需要记录哪个管理员办理了某个读者的借书证信息，…，那么才需要“管理”这个联系。

案例：图书馆管理系统(2)

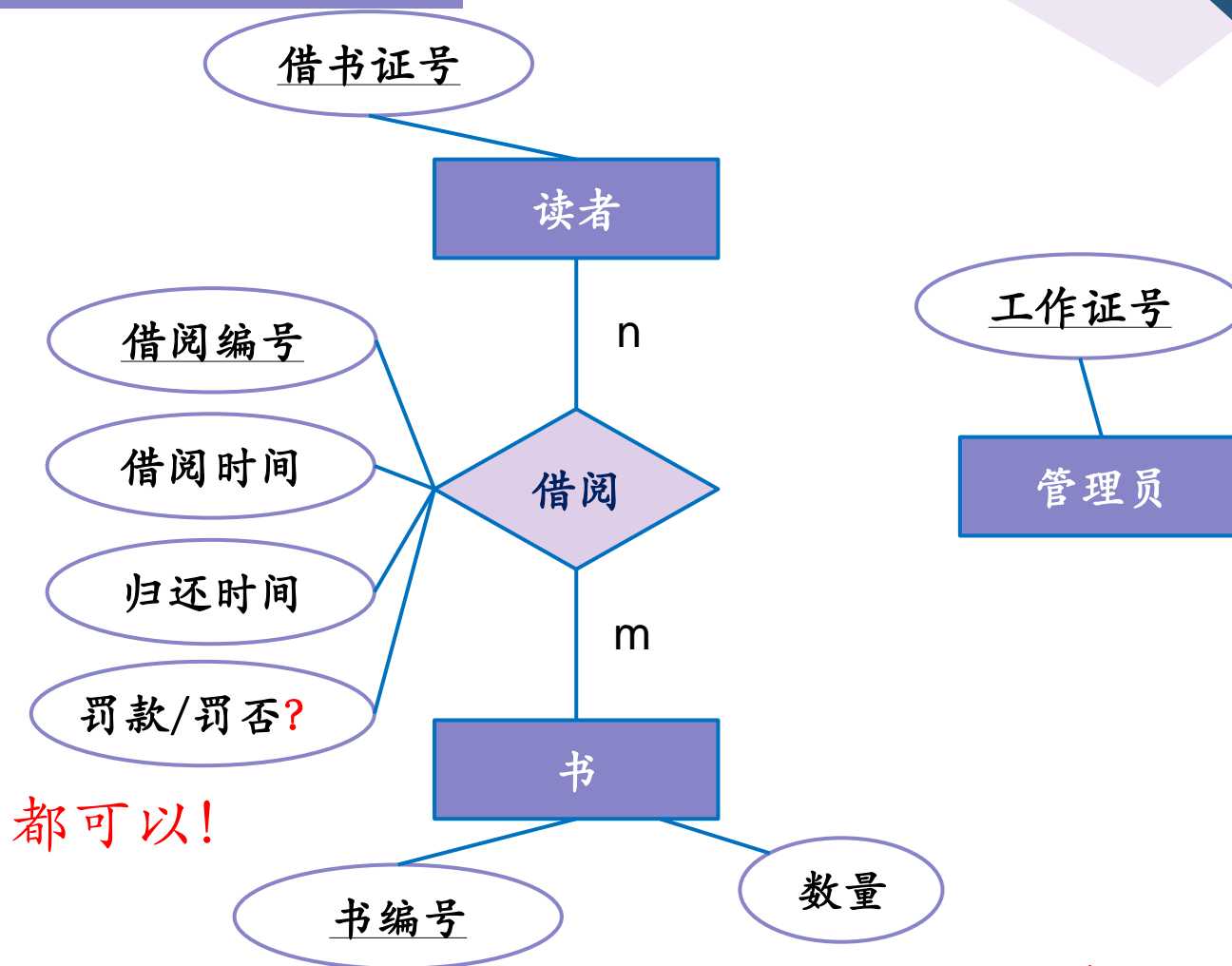
A. 请给出如下场景（满足1~4）的图书管理数据库的ER图设计：

1. 用户类型：读者、管理员

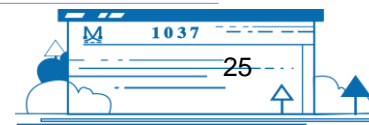
2. 管理员负责图书的管理和读者的管理；

3. 一个读者可以借多本书；一本书可以借给多个读者（馆藏多本）；

4. 一本书最迟1个月要归还；如果逾期还书，需按逾期天数罚款。



满足题意!



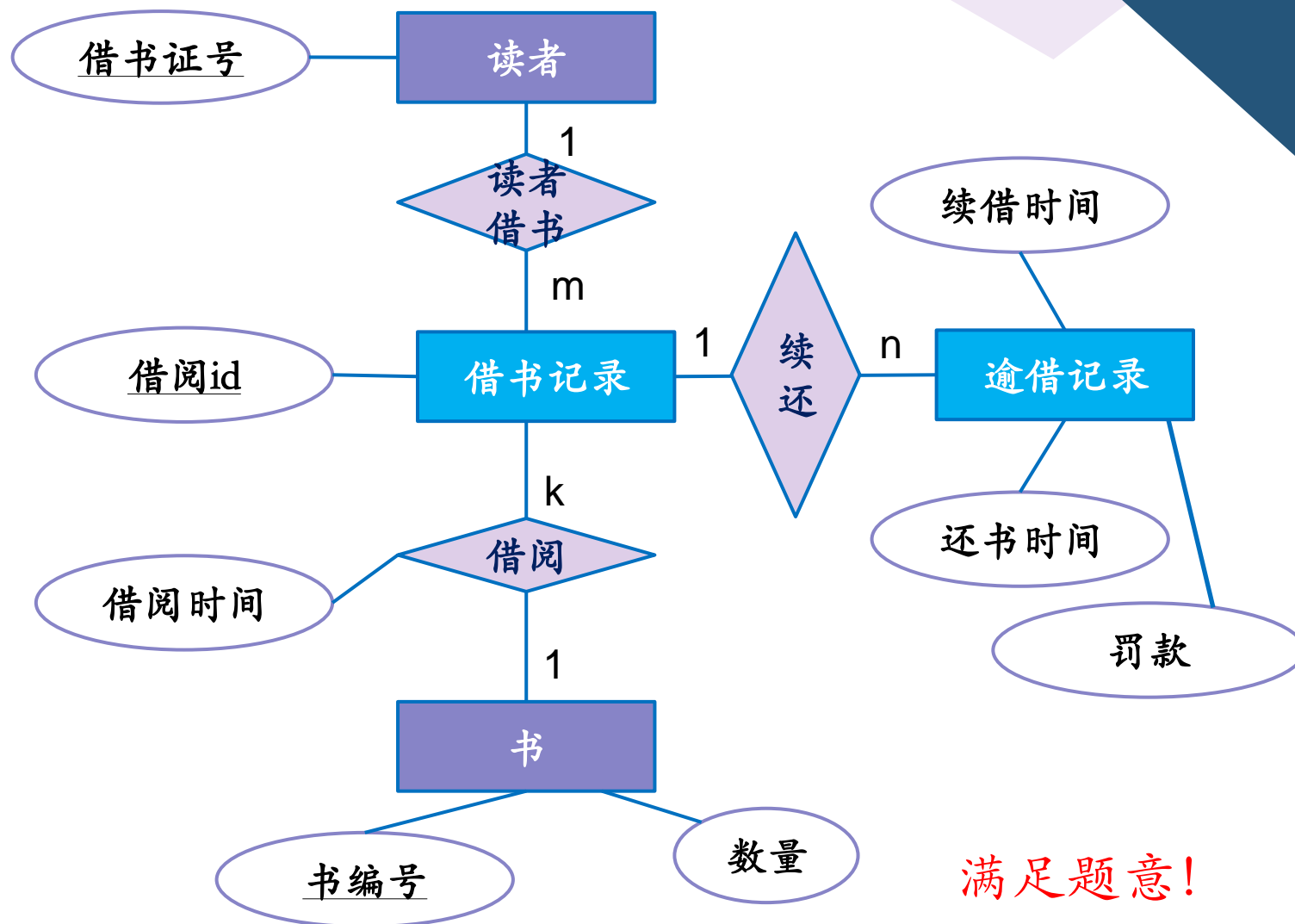
案例：图书馆管理系统(3)

A. 请给出如下场景（满足1~4）的图书管理数据库的ER图设计：

3. 一个读者可以借多本书；一本书可以借给多个读者（馆藏多本）；

4. 一本书最迟1个月要归还；如果逾期还书，需按逾期天数罚款。

□ 借书还书问题的解法2

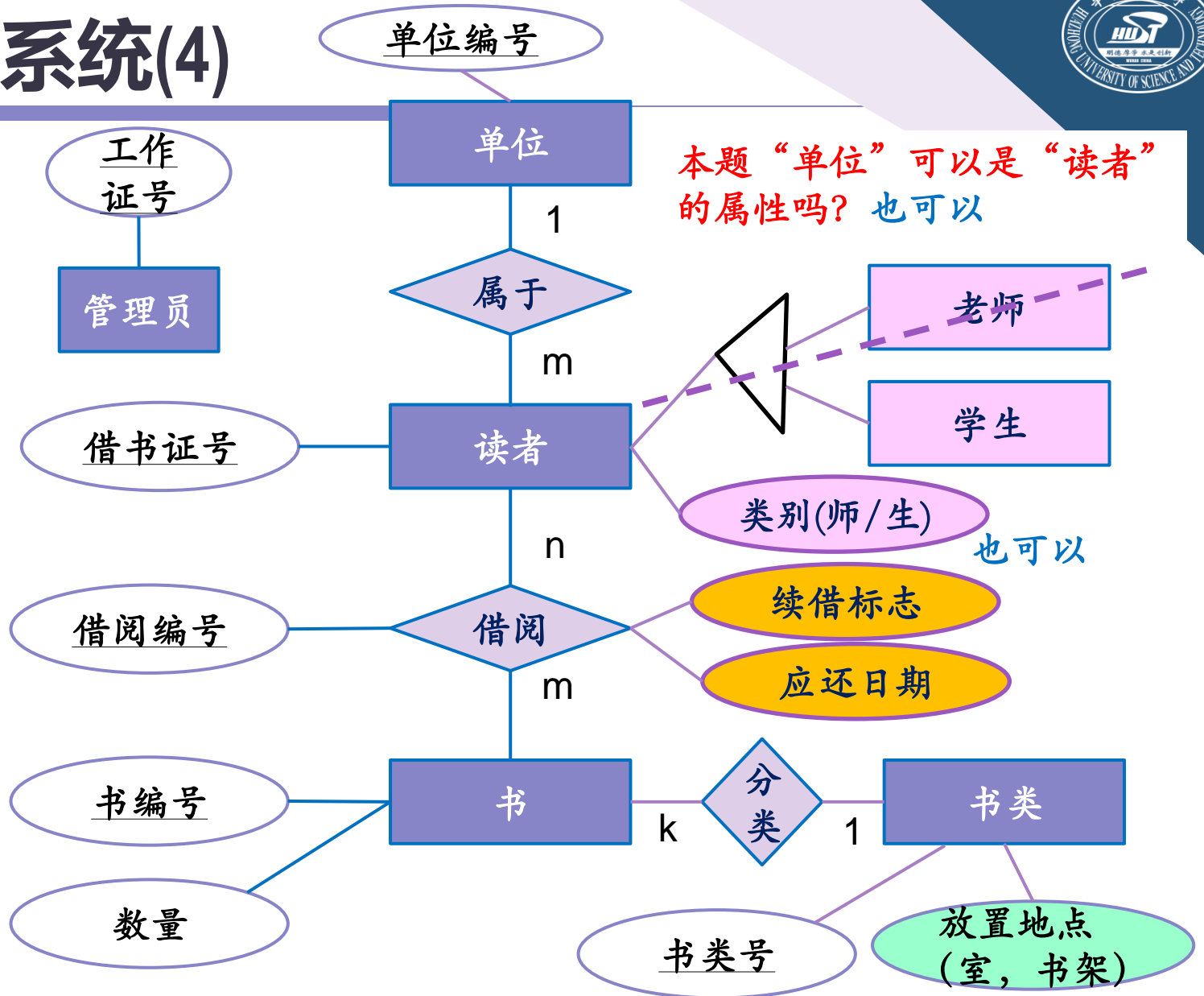


满足题意!

案例：图书馆管理系统(4)

B. 题目升级 (如果要满足5~8, ER图如何设计?) :

1. 读者要有所属单位信息;
2. 图书管分为若干室, 书按图书类型分别放在不同室的不同的书架上;
3. 若第4点改为: 读者可分为: 学生、老师; 学生最多可借有图书5本, 教师最多可以借阅图书10本;
4. 借期30天, 可续借1次, 续借15天; 逾期每册书每天罚款0.2元。

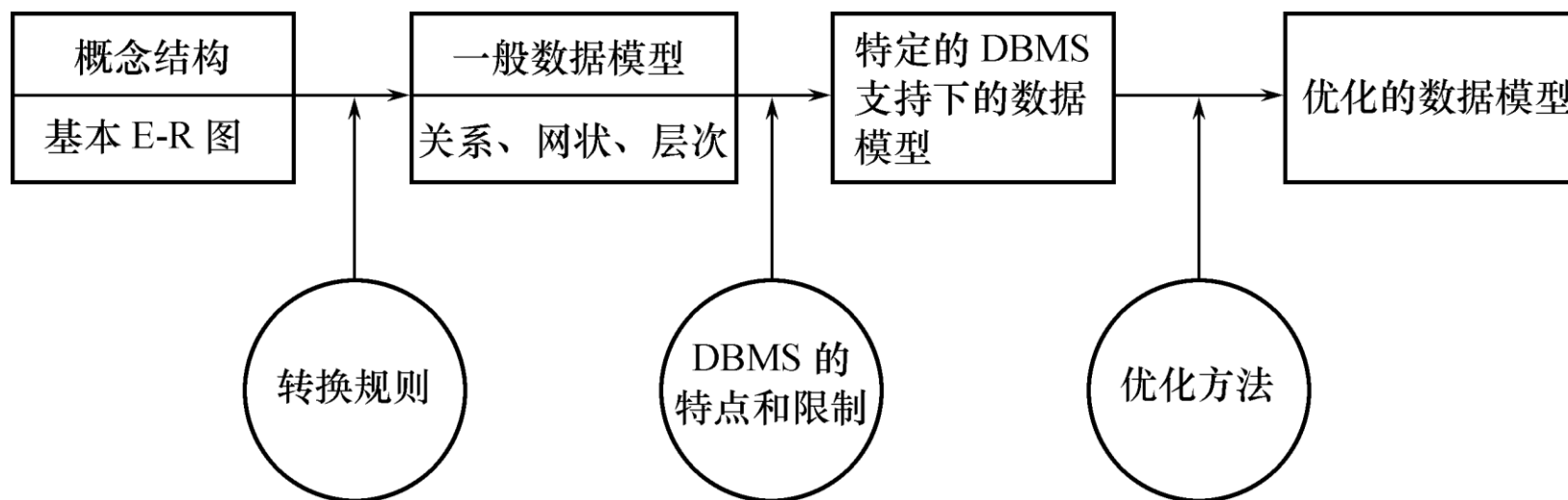


7.4 逻辑结构设计

□ 数据库逻辑设计的任务

- 将概念结构转换成特定DBMS所支持的数据模型。
- 进入“实现设计”阶段，需考虑具体DBMS性能、具体数据模型特点。
- 关系数据库的逻辑设计问题：E-R图如何向关系模型进行转换。

□ 逻辑设计一般分为3个步骤：



7.4.1 E-R图向关系模式的转换

1. 转换原则

(1) 一个实体型转化为一个关系模式

- 关系的属性 = 实体的属性
- 关系的码 = 实体的码

(2) 联系转换方式1:

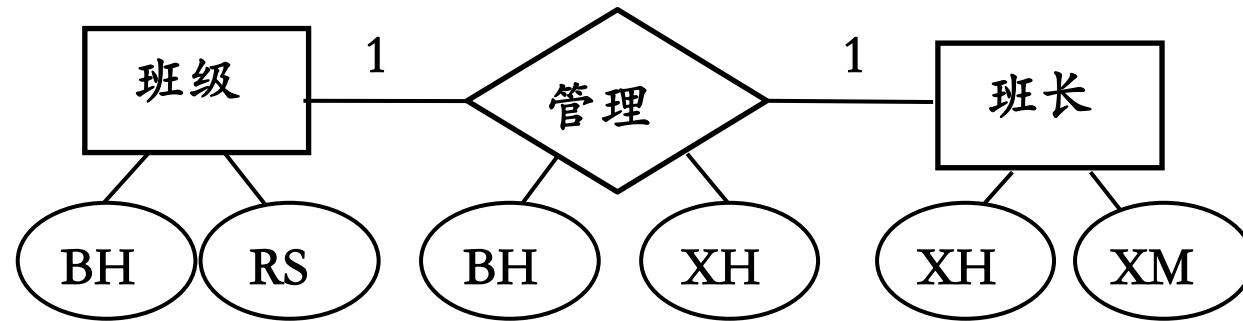
- ① 联系为1:1, 联系实体可消失, 每个实体的键都是关系的候选键;
- ② 联系为1:n, 联系实体可消化到“m”方中去, n端实体的键是关系的键;
- ③ 联系为n:m或多元联系, 联系实体转换为一个关系, 各实体键的组合是关系的键。

联系的转换方式1:

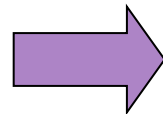
1) 1: 1联系转换为关系模式

- (1) 基本实体 \Rightarrow 一个关系模式;
- (2) 联系实体 \Rightarrow 一个关系模式;
- (3) 联系实体可消化。

例:



班级(BH, RS)
 班长(XH, XM)
 管理(BH, XH)



或

班级 (BH, RS, XH)

班长 (XH, XM)

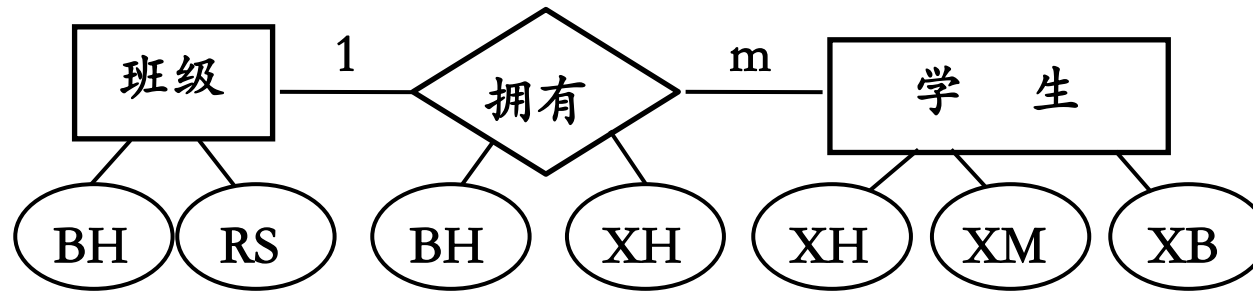
班级 (BH, RS)

班长 (XH, XM, BH)

2) 1: m联系转换为关系模式

- (1) 基本实体 \Rightarrow 一个关系模式;
- (2) 联系实体 \Rightarrow 一个关系模式;
- (3) 联系实体可消化到“m”方中去。

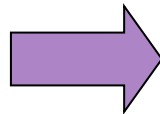
例:



班级 (BH, RS)

学生 (XH, XM, XB)

拥有 (BH, XH)



消化处理后为:

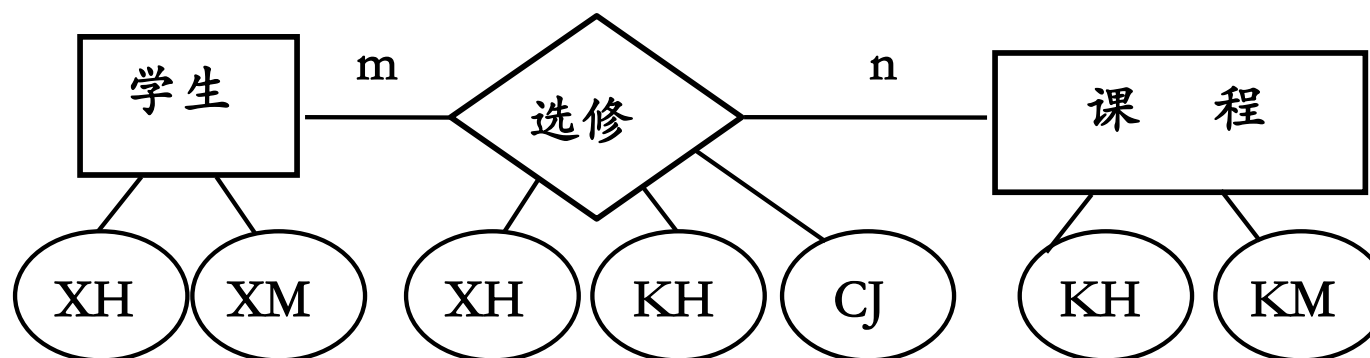
班级(BH, RS)

学生(XH, XM, XB, BH)

3) m: n联系转换为关系模式

- (1) 基本实体 \Rightarrow 一个关系模式;
- (2) 联系实体 \Rightarrow 一个关系模式。

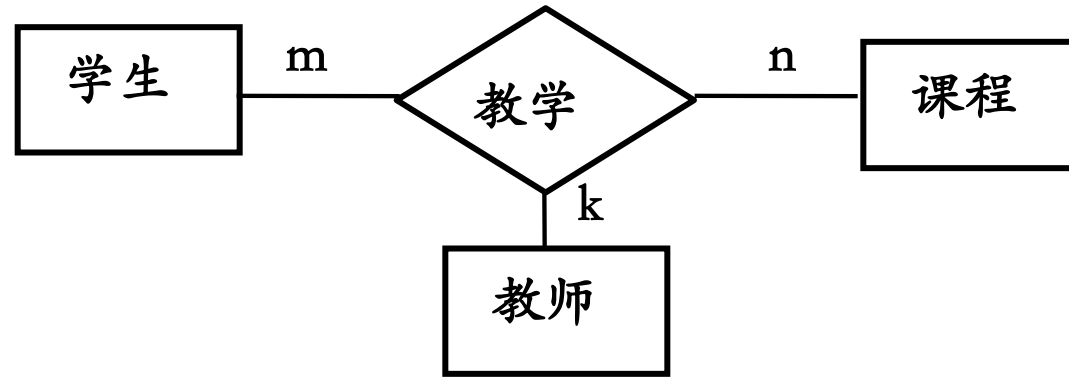
例:



学生 (XH, XM)
课程 (KH, KM)
选修 (XH, KH, CJ)

4) 多元联系转换为关系模式

- (1) 基本实体 \Rightarrow 一个关系模式;
- (2) 联系实体 \Rightarrow 一个关系模式。



学生 (XH,)
课程 (KH,)
教师 (JH,)
教学 (XH, KH, JH)

7.4.1 E-R图向关系模式的转换

1. 转换原则 (续)

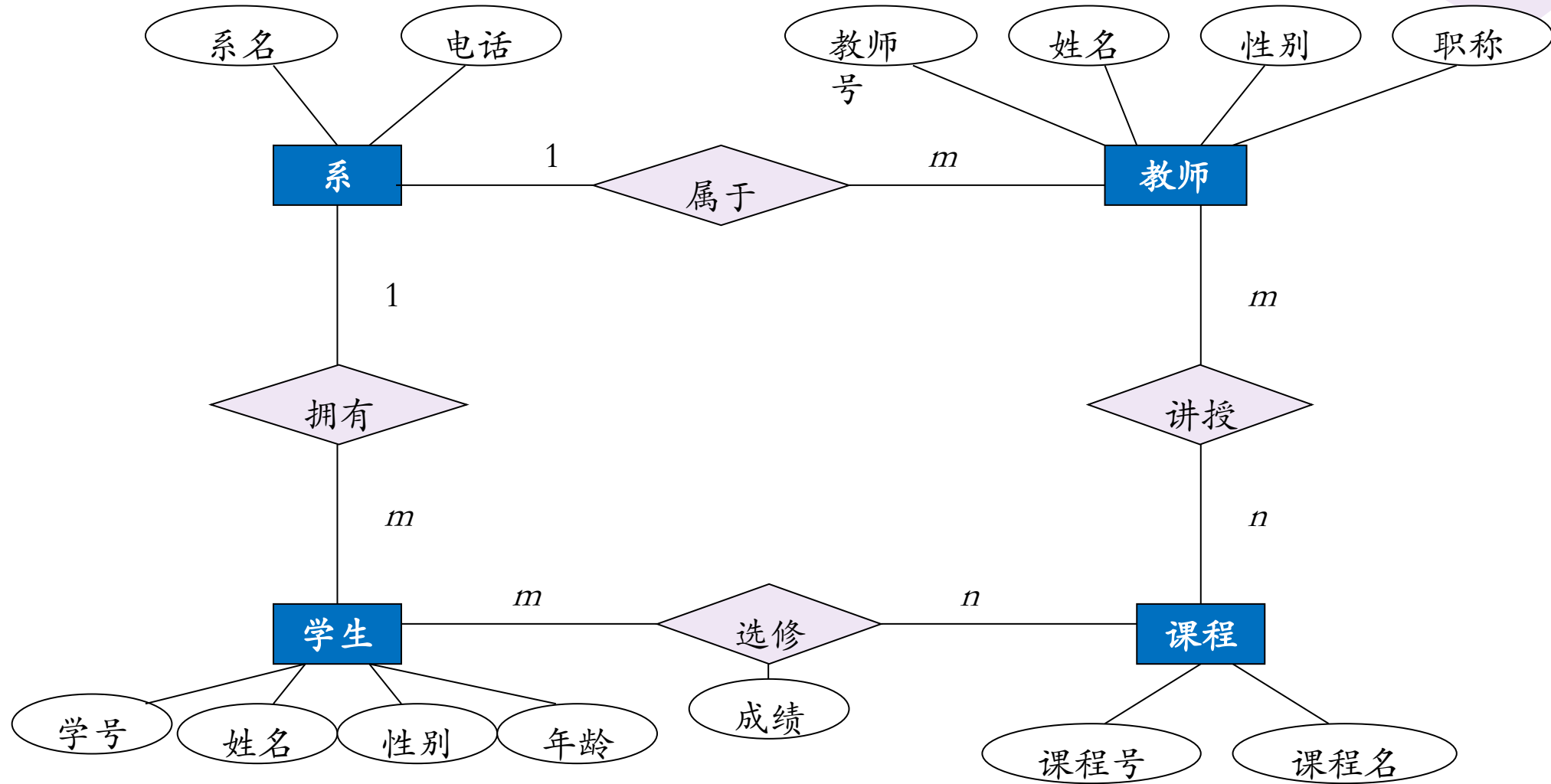
(2') **联系的转换方式2**: 一个联系转换为一个关系模式, 与该联系相连的各实体的键以及联系的属性均转换为该关系的属性。

该关系的键有三种情况:

- ①联系为1:1, 则每个实体的键都是关系的候选键;
- ②联系为1:n, 则n端实体的键是关系的键;
- ③联系为n:m, 则各实体键的组合是关系的键。

(3) 具有相同码的关系模式可以合并。

例：7.3中E-R图例



例：初始关系模式设计

(1) 把每一个实体转换为一个关系

图中的四个实体分别转换成四个关系模式：

学生 (学号, 姓名, 性别, 年龄)

课程 (课程号, 课程名)

教师 (教师号, 姓名, 性别, 职称)

系 (系名, 电话)

(2) 把每一个联系转换为关系模式

图中的四个联系也分别转换成四个关系模式：

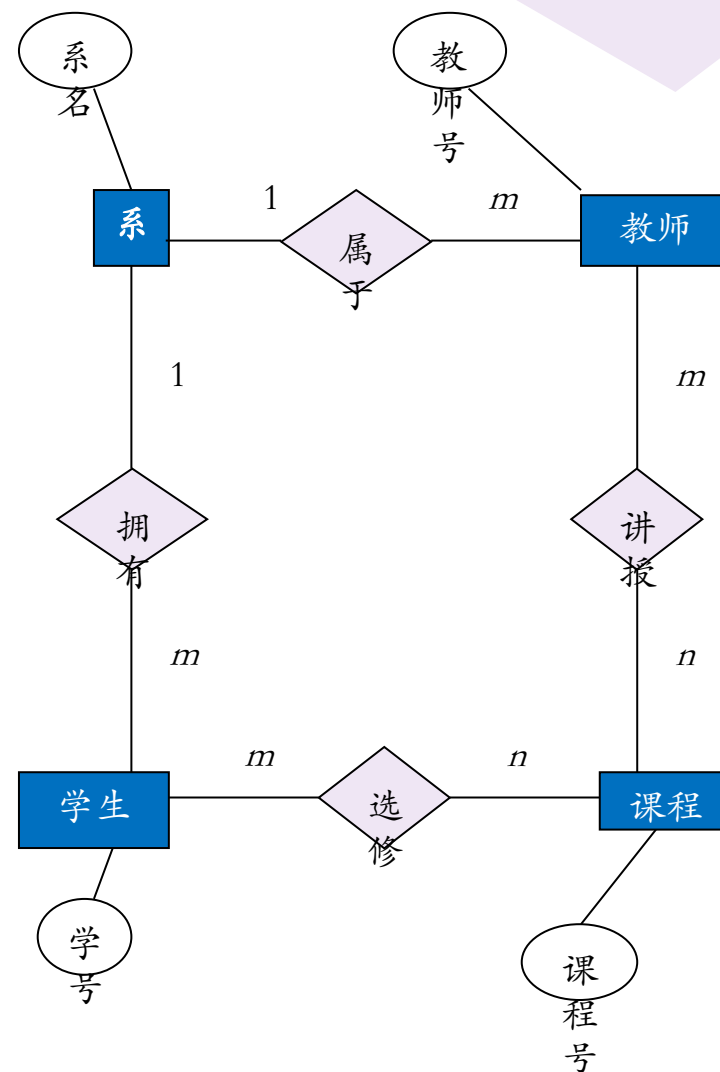
属于 (教师号, 系名)

讲授 (教师号, 课程号)

选修 (学号, 课程号, 成绩)

拥有 (系名, 学号)

其中，有下划线者表示是主键。



优化下列关系模式

□ 优化前:

学生 (学号, 姓名, 性别, 年龄)

课程 (课程号, 课程名)

教师 (教师号, 姓名, 性别, 职称)

系 (系名, 电话)

属于 (教师号, 系名)

讲授 (教师号, 课程号)

选修 (学号, 课程号, 成绩)

拥有 (系名, 学号)

□ 优化后, 学生和教师关系被修改:

学生 (学号, 姓名, 性别, 年龄, 系名)

教师 (教师号, 姓名, 性别, 职称, 系名)

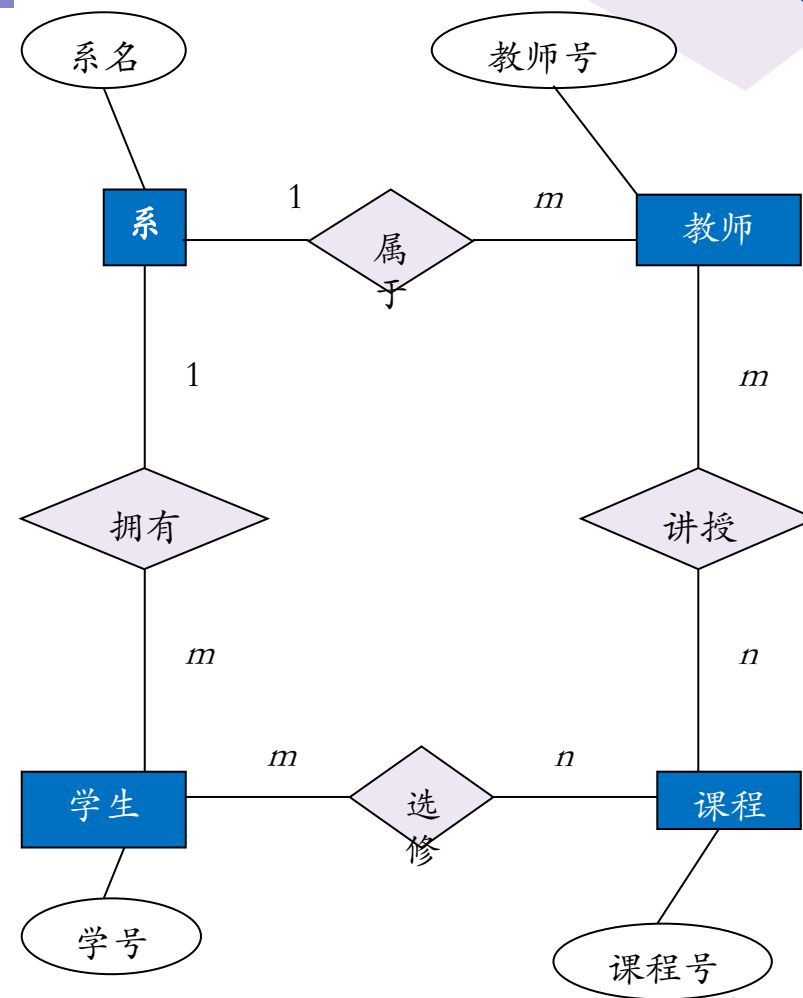
关系模式属于和拥有被消去,
其他关系不变

7.4.2 数据模型的优化

- **数据模型的优化**：数据库逻辑设计的结果**不是唯一的**，得到初步数据模型后，还应该适当地修改、调整数据模型的结构，以进一步提高数据库应用系统的性能。
- 关系数据模型的优化通常以**规范化理论**为指导。
- 优化方法为：
 1. 确定数据依赖
 2. 消除冗余联系
 3. 确定所属范式级别
 4. 检验生成的模式是否合适应用环境实施规范化处理，是否需要某些模式进行合并或分解。并不是规范化程度越高的关系就越优。

7.4.2 数据模型的优化

- 确定函数依赖（第6章知识）：
 - 一对多的联系可表示为：
学号 \rightarrow 系号，教师号 \rightarrow 系号，
 - 多对多的联系可表示为：
(教师号，课程号) \rightarrow 课程细节；
(学生号，课程号) \rightarrow 成绩
- 得到函数依赖集FL
- 求FL的最小覆盖GL，差集为 $D = FL - GL$ 。
逐一考察D中的函数依赖，确定是否是冗余的联系，若是，就把它去掉。



7.4.2 数据模型的优化

4. 对关系模式进行必要分解，提高数据操作效率和存储空间利用率。

1) **水平分解**：把关系的元组分为若干子集合，定义每个子集合为一个子关系。

- 经常进行大量数据的**分类条件查询**的关系，可进行水平分解，可以减少应用系统每次查询需要访问的记录数，从而提高查询性能。

- 适用范围：

- 满足“80/20原则”的应用：20%的常用数据分解出来作为子关系。

- 并发事务经常存取不相交的数据：分解为每个事务存取的数据对应一个关系。

【例】有学生关系（学号，姓名，类别……），其中类别包括大专生、本科生和研究生。如果多数查询一次只涉及其中的一类学生，则按学生类别水平分割整个学生关系。

7.4.2 数据模型的优化

2) **垂直分解**：把关系模式的属性分解为若干子集合，形成若干子关系模式。

■ **原则**：把经常一起使用的属性分解出来，形成一个子关系模式。

【例】有教师关系（教师号，姓名，性别，年龄，职称，工资，岗位津贴，住址，电话），如果经常查询的仅是前六项，而后三项很少使用，则垂直分割：

□ 教师关系1（教师号，姓名，性别，年龄，职称，工资）

□ 教师关系2（教师号，岗位津贴，住址，电话）

■ **优点**：提高某些事务的效率

■ **缺点**：也有可能使另一些事务不得不执行连接操作，从而降低了效率。

■ **衡量标准**：分解后的所有事务的总效率是否得到了提高？还要保证分解后的关系具有无损连接性和函数依赖保持性。

7.4.3 设计用户子模式

根据局部应用需求，结合具体DBMS的特点设计用户外模式，即关系DBMS提供的视图。

1. 指导思想

注重局部用户的习惯与方便。

2. 具体方法

- 使用更符合用户习惯的别名；
- 对不同级别的用户定义不同的视图，以保证系统的安全性；
- 简化用户对系统的使用。

如：某些用户的复杂查询，涉及多个关系模式，可为其定义专门视图，大大地简化用户地使用。

设计用户子模式

[例] 关系模式：产品（产品号，产品名，规格，单价，生产车间，生产负责人，产品成本，产品合格率，质量等级），可以在产品关系上建立多个视图：

- 顾客视图中只包含允许顾客查询的属性
- 销售部门视图中只包含允许销售部门查询的属性
- 生产领导部门则可以查询全部产品数据

如：

- 为一般顾客建立视图： 产品1（产品号，产品名，规格，单价）
- 为产品销售部门建立视图：
产品2（产品号，产品名，规格，单价，车间，生产负责人）

7.5 物理结构设计

- 对于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，称为**数据库物理设计**。
- 任务：以逻辑设计的结果为输入，结合具体DBMS的特点与存储设备特性进行设计，选定数据库在物理设备上的存储结构和存取方法。
- 数据库的物理设计可分为两步：
 - (1) **确定物理结构**：在关系数据库中主要指**存取方法和存储结构**；
 - (2) **评价物理结构**：评价的重点是**时间和空间效率、维护代价和各种用户要求进行权衡**，其结果可以产生多种方案，数据库设计人员必须对这些方案进行细致的评价，从中选择一个较优的方案。

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：

- 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

- **数据查询事务**：关系、查询条件所涉及的属性、连接条件所涉及的属性，查询投影属性；

- **数据更新事务**：关系、更新条件所涉及所有属性，每个事务的频率和性能要求；

- 充分了解所用RDBMS的内部特征，特别是系统提供的**存取方法**（建立存取路径）和 **存储结构**（关系、索引等）

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**。

- 存取方法的设计

存取方法是为存储在物理设备（通常指辅存）上的数据提供存储和检索能力的方法。常用的存取方法包括聚簇和索引。

- (1) 索引：经典存取方法 - B+树索引
- (2) 聚簇 (Cluster)
- (3) HASH方法

1. 索引

- 索引是根据某些属性的值建立的值与对应记录的存储位置间的映射关系表，它可以提供对数据的**快速访问**。

- 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**？

1. 索引

□ 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

2. 聚簇

□ **聚簇**：为提高某个属性（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。**

□ 聚簇有两个作用：

① 聚簇码相同的元组集中在一起，因而聚簇值不必在每个元组中重复存储，只要在一**组中存储一次**即可，因此可以节省存储空间。

② 聚簇功能可以大大提高按聚簇码进行查询的效率。

【例】假设要查询学生关系中计算机系的学生名单（300名学生）：

■ 极端情况下，记录分布在300个不同的物理块中，查询计算机系的学生，要做300次的I/O操作，将影响系统查询的性能。

■ 若按照系别建立聚簇，使同一个系的学生记录集中存放，则每做一次I/O操作，就可获得多个满足查询条件和记录，从而显著地减少了访问磁盘的次数。

2. 聚簇

□ 设计候选聚簇：

- 对经常在一起进行连接操作的关系可以建立聚簇；
- 如果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；
- 如果一个关系的一个(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。太少则聚簇的效果不明显。

□ 优化聚簇设计：

- 从聚簇中删除经常进行全表扫描的关系；
- 从聚簇中删除更新操作远多于连接操作的关系；
- 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。

2. 聚簇

□ 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，这一查询涉及学生关系和选修关系的连接操作，即需要按学号连接这两个关系：

为提高连接操作的效率，可以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。

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。

尤其当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。

2. 聚簇

□ 聚簇的局限性：

1. 聚簇只能提高某些特定应用的性能。
2. 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。

3. HASH存取

□ 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相等比较选择条件中；

□ 该关系的大小可预知，而且不变；

或：

□ 该关系大小动态改变，但所选用的DBMS提供了动态HASH存取方法

7.5.3 确定数据库的存储结构

1. 确定数据的存放位置

为了提高系统性能，应该根据应用情况将数据的易变部分、稳定部分、经常存取部分和存取频率较低部分分开存放。

- 例如：许多计算机都有多个磁盘，因此可以将表和索引分别存放在不同的磁盘上，在查询时，由于两个磁盘驱动器并行工作，可以提高物理读写的速度。
- 在多用户环境下，可以将日志文件和数据库对象（表、索引等）放在不同的磁盘上，以加快存取速度。
- 另外，数据库的数据备份、日志文件备份等，只在数据库发生故障进行恢复时才使用，而且数据量很大，可以存放在磁带上，以改进整个系统的性能。

7.5.3 确定数据库的存储结构

2. 确定系统配置

- DBMS产品一般都提供了一些系统配置变量、存储分配参数，供设计人员和DBA对数据库进行物理优化。系统为这些变量设定了初始值，但是这些值不一定适合每一种应用环境，在物理设计阶段，要根据实际情况重新对这些变量赋值，以满足新的要求。
- 系统配置变量和参数很多，例如，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、数据库的大小、时间片的大小、锁的数目等，这些参数值影响存取时间和存储空间的分配，在物理设计时要根据应用环境确定这些参数值，以使系统的性能达到最优。

7.5.4 评价物理结构

- 在确定了数据库的物理结构之后，要进行评价，重点是时间和空间的效率。
- 评价方法（完全依赖于所选用的DBMS）
 - 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

7.6 数据库实施和维护

- 数据库实施是指根据逻辑设计和物理设计的结果，在计算机上建立起实际的数据库结构、装入数据、进行测试和试运行的过程。

- 数据库实施主要包括以下工作：
 - 建立实际数据库结构；
 - 装入数据；
 - 数据库试运行；
 - 整理文档。

1. 建立实际数据库结构

- ◆ DBMS提供的数据库定义语言（DDL）可以定义数据库结构。可使用SQL定义语句中的CREATE TABLE语句定义所需的基本表，使用CREATE VIEW语句定义视图。

2. 载入数据

- ◆ 装入数据又称为**数据库加载 (Loading)**，是数据库实施阶段主要工作。在数据库结构建立好之后，就可以向数据库中加载数据了。
- ◆ 由于数据库的数据量一般都很大，它们分散于一个企业（或组织）中各个部门的数据文件、报表或**多种形式的**单据中，它们存在着**大量的重复**，并且其格式和结构一般都不符合数据库的要求，必须把这些**数据收集**起来加以**整理**，**去掉冗余并转换成数据库所规定的格式**，这样处理之后才能装入数据库。因此，需要耗费大量的人力、物力，是一种非常单调乏味而又意义重大的工作。

载入

- 由于应用环境和数据来源的差异，所以不可能存在普遍通用的转换规则，现有的DBMS并不提供通用的数据转换软件来完成这一工作。
- 对于一般的小型系统，装入数据量较少，可以采用人工方法来完成。
 - 首先将需要装入的数据从各个部门的数据文件中筛选出来，转换成符合数据库要求的数据格式，
 - 然后输入到计算机中，
 - 最后进行数据校验，检查输入的数据是否有误。
- 但是，人工方法不仅效率低，而且容易产生差错。对于数据量较大的系统，应该由计算机来完成这一工作。通常是设计一个数据输入子系统，其主要功能是从大量的原始数据文件中筛选、分类、综合和转换数据库所需的数据，把它们加工成数据库所要求的结构形式，最后装入数据库中。

校验

- 为了保证装入数据库中数据的正确无误，必须高度重视数据的校验工作。在**输入子系统**的设计中应该考虑**多种数据检验技术**，在数据转换过程中应使用不同的方法进行多次检验，确认正确后方可入库。
- 如果在数据库设计时，原来的数据库系统仍在运行，则数据的转换工作是将原来老系统中的数据转换成新系统中的数据结构。同时还要转换原来的应用程序，使之能在新系统下有效地运行。
- 数据的转换、分类和综合常常需要多次才能完成，因而输入子系统的设计和实现是很复杂的，需要编写许多应用程序，由于这一工作需要耗费较多的时间，为了保证数据能够及时入库，应该在**数据库物理设计的同时编制数据输入子系统，而不能等物理设计完成后才开始。**

7.6.2 数据库试运行

应用程序编写完成，并有了一小部分数据装入后，应该按照系统支持的各种应用分别试验应用程序在数据库上的操作情况，这就是数据库的试运行阶段，或者称为联合调试阶段。在这一阶段要完成两方面的工作。

- (1) **功能测试**。实际运行应用程序，测试它们能否完成各种预定的功能。
- (2) **性能测试**。测量系统的性能指标，分析系统是否符合设计目标。

系统的试运行对于系统设计的性能检验和评价是很重要的，因为有些参数的最佳值只有在试运行后才能找到。如果测试的结果不符合设计目标，则应返回到设计阶段，重新修改设计和编写程序，有时甚至需要返回到逻辑设计阶段，调整逻辑结构。

7.6.2 数据库试运行

- ◆ 重新设计物理结构甚至逻辑结构，会导致数据重新入库。由于数据装入的工作量很大，所以可分期分批的组织数据装入，先输入小批量数据做调试用，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。
- ◆ 数据库的实施和调试不是几天就能完成的，需要有一定的时间。在此期间由于系统还不稳定，随时可能发生硬件或软件故障，加之数据库刚刚建立，操作人员对系统还不熟悉，对其规律缺乏了解，容易发生操作错误，这些故障和错误很可能破坏数据库中的数据，这种破坏很可能在数据库中引起连锁反应，破坏整个数据库。
- ◆ 因此必须做好**数据库的转储和恢复工作**，要求设计人员熟悉DBMS的转储和恢复功能，并根据调试方式和特点首先加以实施，尽量减少对数据库的破坏，并简化故障恢复。

整理文档

- 在程序的编码调试和试运行中，应该将发现的问题和解决方法记录下来，将它们整理存档作为资料，供以后正式运行和改进时参考。
- 全部调试工作完成后，应该编写应用系统的技术说明书和使用说明书，在正式运行时随系统一起交给用户。
- 完整的文件资料是应用系统的重要组成部分，但这一点常被忽视。必须强调这一工作的重要性，引起用户与设计人员的充分注意。

7.6.3 数据库运行和维护

- 数据库试运行结果符合设计目标后，数据库就投入正式运行，进入运行和维护阶段。数据库系统投入正式运行，标志着数据库应用开发工作的基本结束，但并不意味着设计过程已经结束。
- 由于应用环境不断发生变化，用户的需求和处理方法不断发展，数据库在运行过程中的存储结构也会不断变化，从而必须修改和扩充相应的应用程序。
- 数据库运行和维护阶段的主要任务包括以下三项内容：
 - (1) 维护数据库的安全性与完整性
 - (2) 监测并改善数据库性能
 - (3) 重新组织和构造数据库

1. 维护数据库的安全与完整性

- 按照设计阶段提供的安全规范和故障恢复规范，DBA要经常检查系统的安全是否受到侵犯，**根据用户的实际需要授予用户不同的操作权限。**
- 数据库在运行过程中，由于应用环境发生变化，对安全性的要求可能发生变化，**DBA要根据实际情况及时调整相应的授权和密码，以保证数据库的安全性。**
- 同样数据库的**完整性约束条件也可能会随应用环境的改变而改变**，这时DBA也要对其进行调整，以满足用户的要求。
- 另外，为了确保系统在发生故障时，能够及时地进行恢复，DBA要针对不同的应用要求**定制不同的转储计划**，定期对数据库和日志文件进行备份，以使数据库在发生故障后恢复到某种一致性状态，保证数据库的完整性。

2. 监测并改善数据库性能

- 目前许多DBMS产品都提供了**监测系统性能参数的工具**，DBA可以利用系统提供的这些工具，经常对数据库的**存储空间状况及响应时间**进行分析评价；
- 结合用户的反应情况确定改进措施；及时改正运行中发现的错误；
- 按用户的要求对数据库的现有功能进行适当的扩充。
- 但要注意在增加新功能时应保证原有功能和性能不受损害。

3. 重新组织和构造数据库

- 数据库建立后，除了数据本身是动态变化以外，随着应用环境的变化，数据库本身也必须变化以适应应用要求。
- 数据库运行一段时间后，由于记录的不断增加、删除和修改，会改变数据库的物理存储结构，使数据库的物理特性受到破坏，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。因此，需要对数据库进行重新组织，即重新安排数据的存储位置，回收垃圾，减少指针链，改进数据库的响应时间和空间利用率，提高系统性能。这与操作系统对“磁盘碎片”的处理的概念相类似。
- 数据库的重组只是使数据库的物理存储结构发生变化，而数据库的逻辑结构不变，所以根据数据库的三级模式，可以知道数据库重组对系统功能没有影响，只是为了提高系统的性能。

3. 重新组织和构造数据库

- 数据库应用环境的变化可能导致数据库的逻辑结构发生变化，比如要增加新的实体，增加某些实体的属性，这样实体之间的联系发生了变化，这样使原有的数据库设计不能满足新的要求，**必须对原来的数据库重新构造**，适当调整数据库的模式和内模式，比如要增加新的数据项，增加或删除索引，修改完整性约束条件等。
- DBMS一般都提供了重新组织和构造数据库的应用程序，以帮助DBA完成数据库的重组和重构工作。
- 只要数据库系统在运行，就需要不断地进行修改、调整和维护。一旦应用变化太大，数据库重新组织也无济于事，这就表明数据库应用系统的生命周期结束，应该建立新系统，重新设计数据库。从头开始数据库设计工作，标志着一个新的数据库应用系统生命周期的开始。

本章小结

□ 数据库的设计过程:

- 需求分析
- 概念结构设计: 概念模型: E-R图
- 逻辑结构设计: 逻辑模式、外模式
- 物理设计: 内模式
- 实施和维护

考点*: 给定一个数据库应用场景, 请给出相关的ER图设计和数据库表设计。

拓展思考: 如果你是一个DBA, 面对一个具体的数据库应用, 如何设计相应的DB物理结构呢? 从哪些角度来设计?

□ 本章作业: P239 7, 9, 11 (7) , 12

课堂练习

(2001华工计算机专业考研试题): 某运动会历经数届, 每届在不同地点举行, 设有多项体育项目比赛, 有若干运动队参加, 每个运动队有许多运动员, 一个运动员仅可为一个运动队的队员, 可参加多届运动会的多个体育项目的比赛。要建立一个数据库完成下列任务:

1. 登记和检索历届运动会参与的运动队和运动员信息;
2. 登记和检索运动员参加历届运动会的各种比赛项目及成绩。

请完成下述工作:

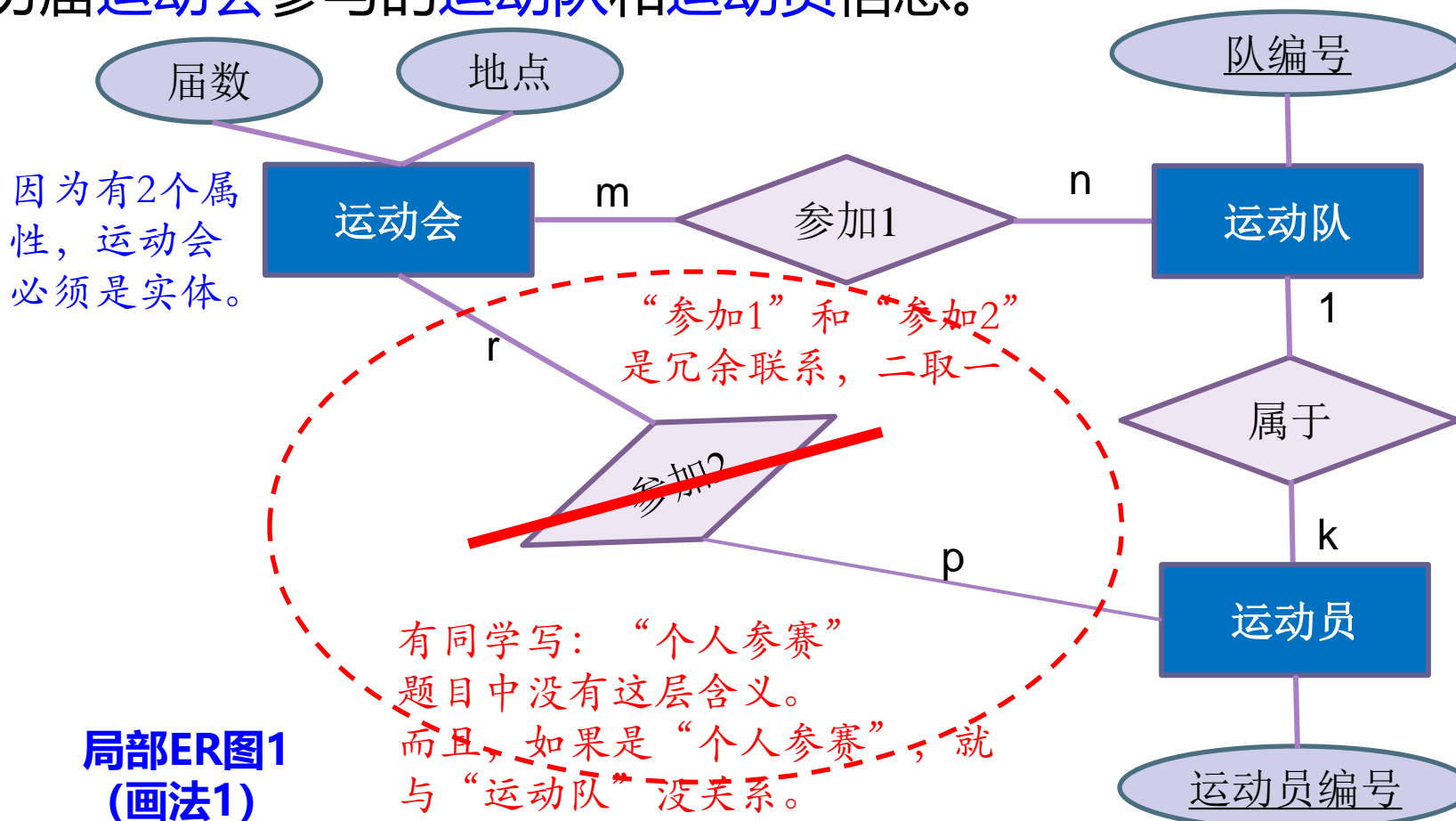
1. 按通常语义拟定实体, 属性后分别构造上述应用的局部ER图
2. 合并它们为全局ER图并说明合并应注意哪些问题 (哪些冲突) ?
3. 将你的ER图转换为关系模式, 并标明关键字。给出建表语句。

课堂练习题

1. 按通常语义拟定实体，属性后分别构造上述应用的**局部ER图**

(1) 登记和检索历届**运动会**参与的**运动队**和**运动员**信息。

某**运动会**历经数**届**，每届在不同**地点**举行，设有**多项体育项目比赛**，有若干**运动队**参加，每个运动队有许多**运动员**，一个**运动员**仅可为一个**运动队的队员**，可参加多届运动会的多个体育项目的比赛。



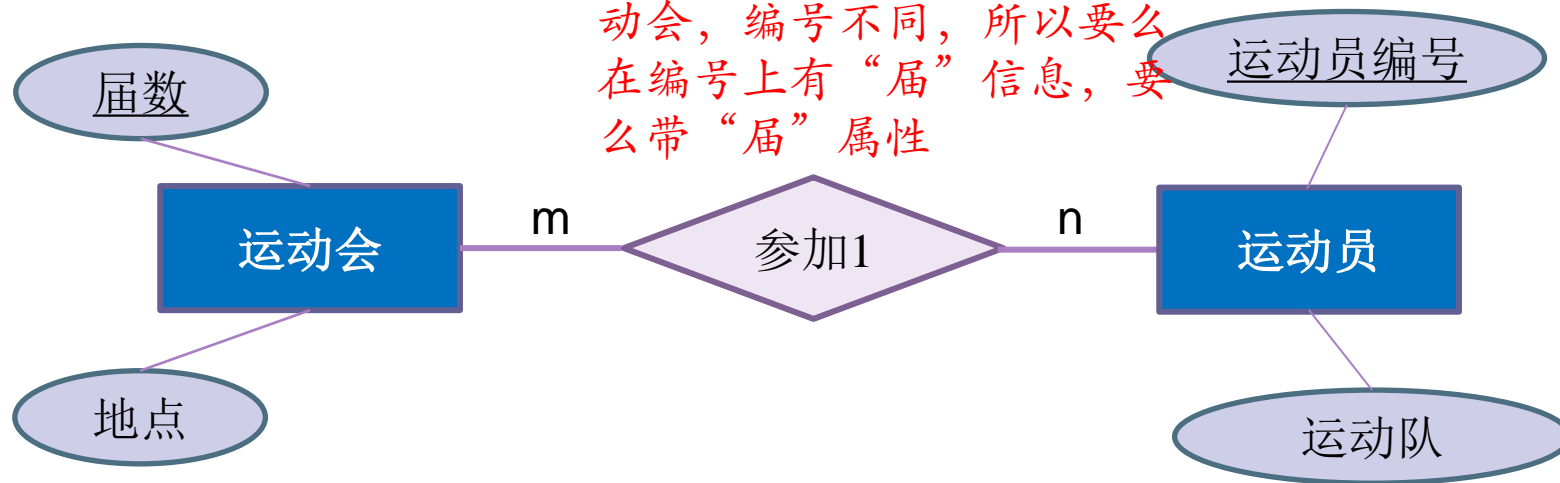
课堂练习题

1. 按通常语义拟定实体，属性后分别构造上述应用的**局部ER图**

(1) 登记和检索历届**运动会**参与的**运动队**和**运动员**信息。

某**运动会**历经数**届**，每届在不同**地点**举行，设有**多项体育项目比赛**，有若干**运动队**参加，每个运动队有许多**运动员**，一个**运动员**仅可为一个**运动队的队员**，可参加多届**运动会的多个体育项目的比赛**。

一个运动员可能参加多次运动会，编号不同，所以要么在编号上有“届”信息，要么带“届”属性



**局部ER图1
(画法2)**

这种画法本题也是正确的。

“运动队”能不能为属性，本题语义没给出判断标准。考虑可否表达以下语义，如：

“运动员A 是作为甲队的成员参加了第1届运动会；而 A 作为乙队的成员参加了第2届运动会。”

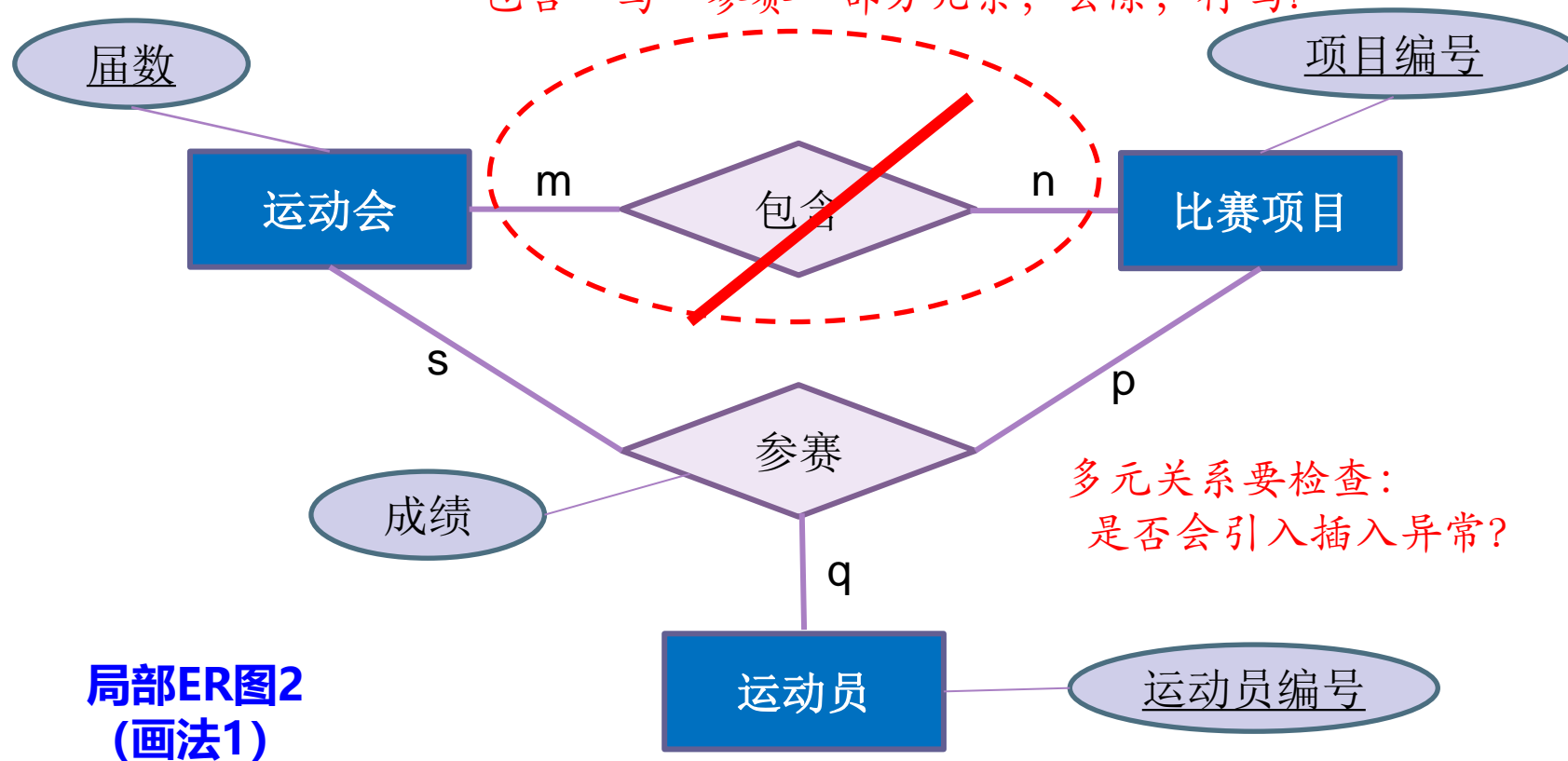
课堂练习题

1. 按通常语义拟定实体，属性后分别构造上述应用的**局部ER图**

(2) 登记和检索**运动员**参加历届**运动会**的各种**比赛项目**及**成绩**。

某**运动会**历经数**届**，每届在不同**地点**举行，设有**多项体育项目比赛**，有若干**运动队**参加，每个运动队有许多**运动员**，一个运动员仅可为一个运动队的队员，可**参加多届运动会**的多个体育项目的比赛。

“包含”与“参赛”部分冗余，去除，行吗？



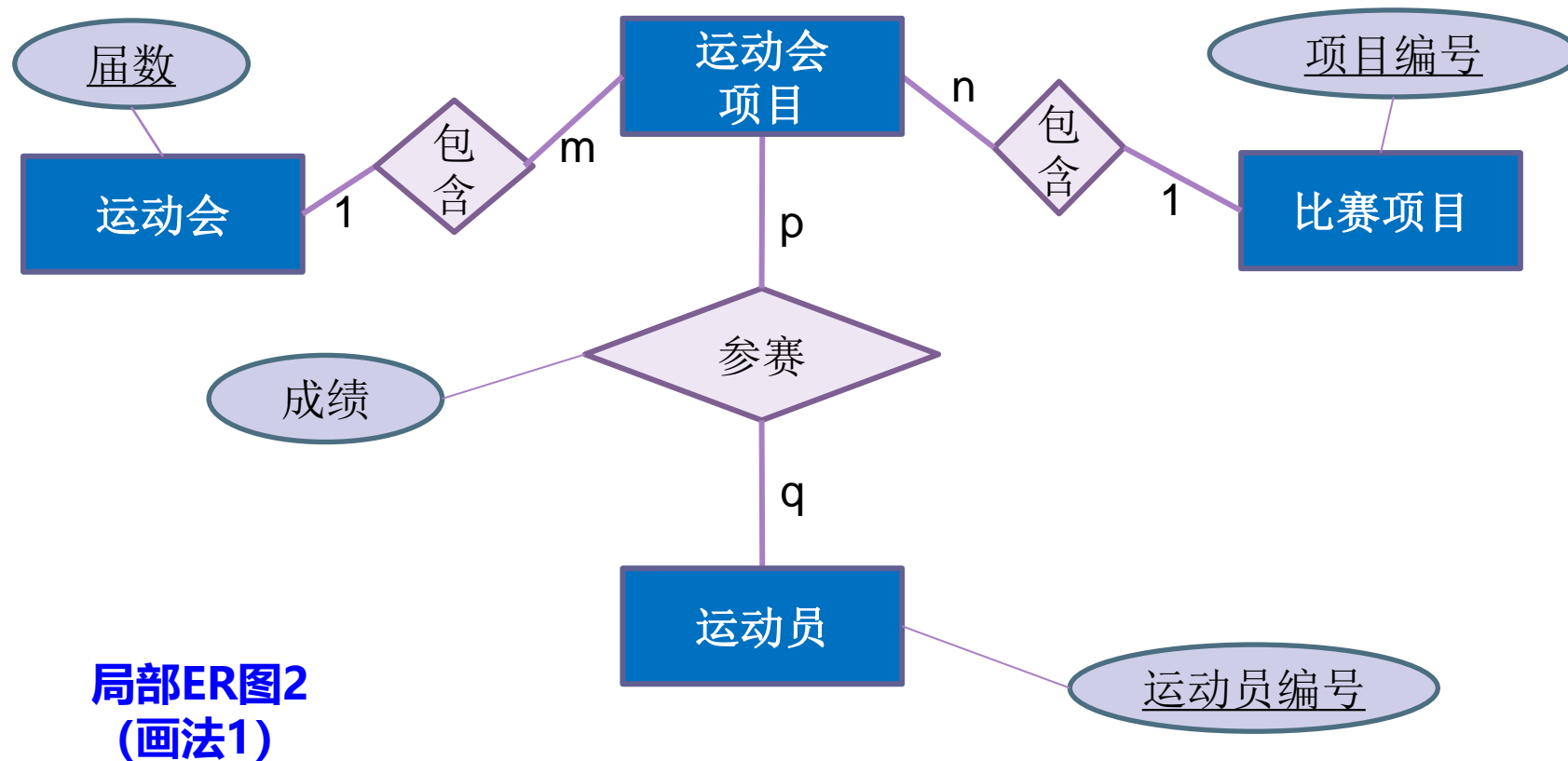
多元关系要检查：
是否会引入插入异常？

课堂练习题

1. 按通常语义拟定实体，属性后分别构造上述应用的局部ER图

(2) 登记和检索运动员参加历届运动会的各种比赛项目及成绩。

某运动会历经数届，每届在不同地点举行，设有多个体育项目比赛，有若干运动队参加，每个运动队有许多运动员，一个运动员仅可为一个运动队的队员，可参加多届运动会的多个体育项目的比赛。

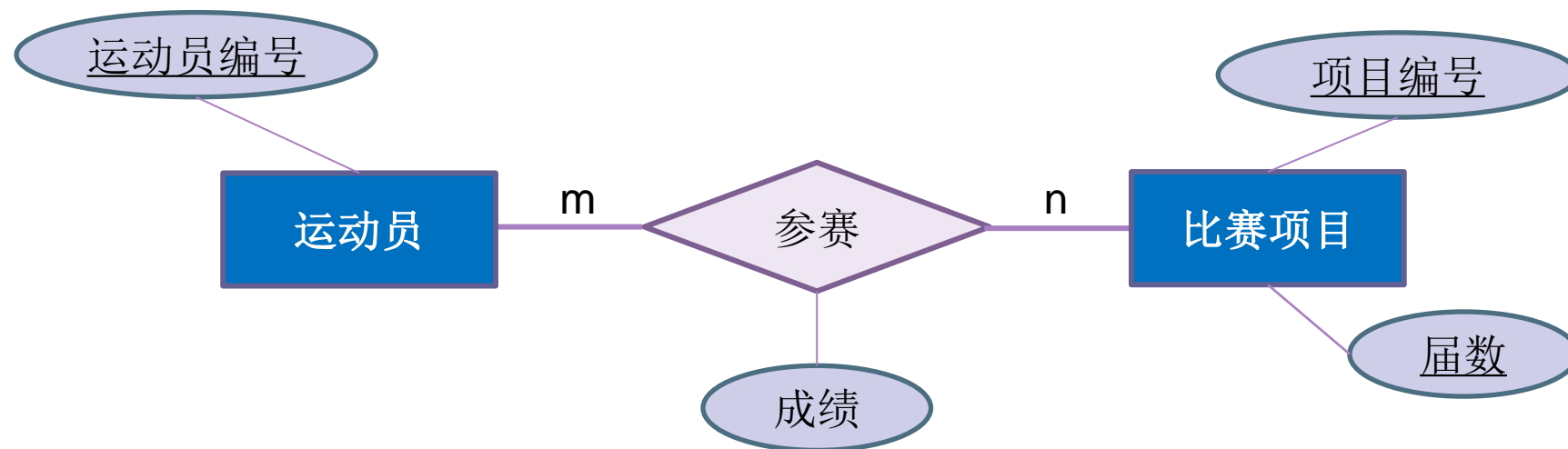


课堂练习题

1. 按通常语义拟定实体，属性后分别构造上述应用的局部ER图

(2) 登记和检索运动员参加历届运动会的各种比赛项目及成绩。

某运动会历经数届，每届在不同地点举行，设有多个体育项目比赛，有若干运动队参加，每个运动队有许多运动员，一个运动员仅可为一个运动队的队员，可参加多届运动会的多个体育项目的比赛。



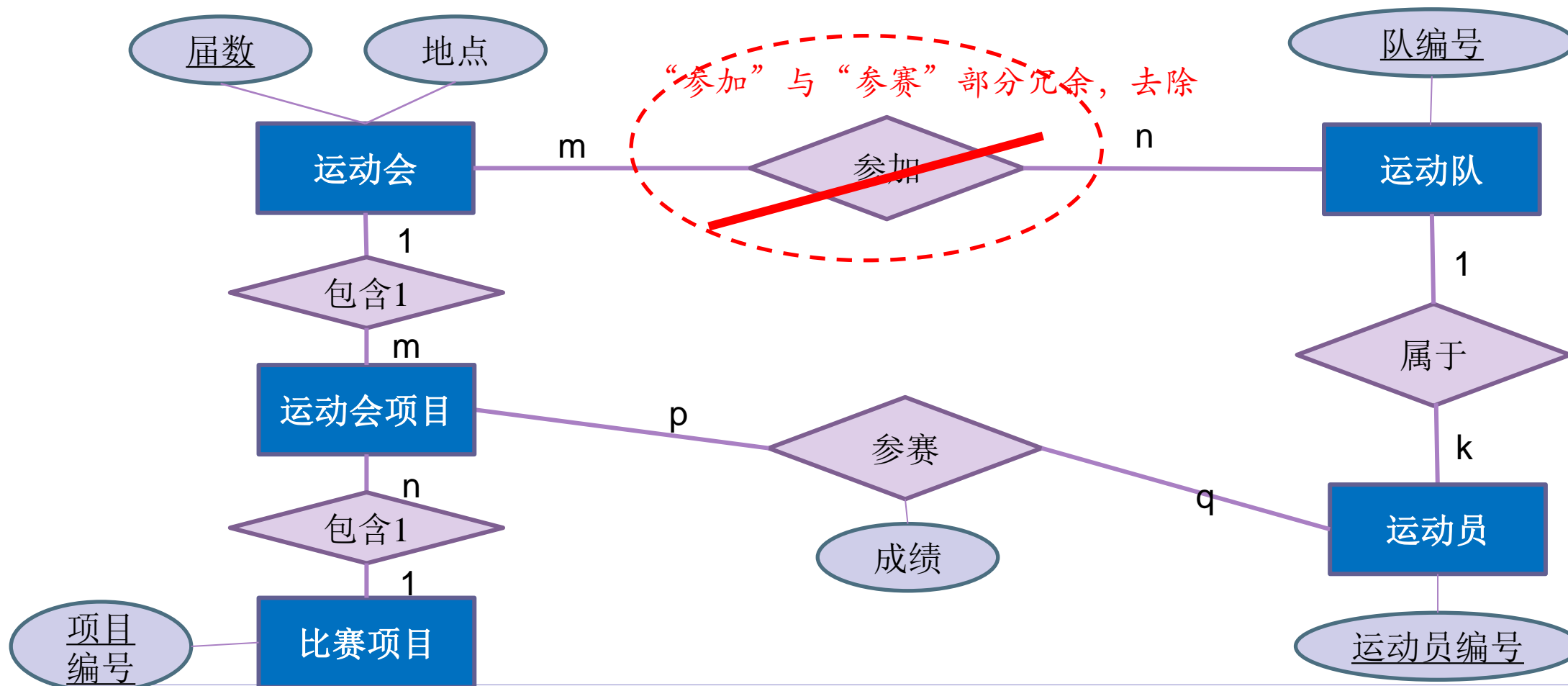
局部ER图2
(画法2)

作为局部ER图，这种画法也是正确的。
但合并时，会存在有关运动会的“结构冲突”。

课堂练习题

2. 全局ER图。

某运动会历经数届，每届在不同地点举行，设有多项体育项目比赛，有若干运动队参加，每个运动队有许多运动员，一个运动员仅可为一个运动队的队员，可参加多届运动会的多个体育项目的比赛。



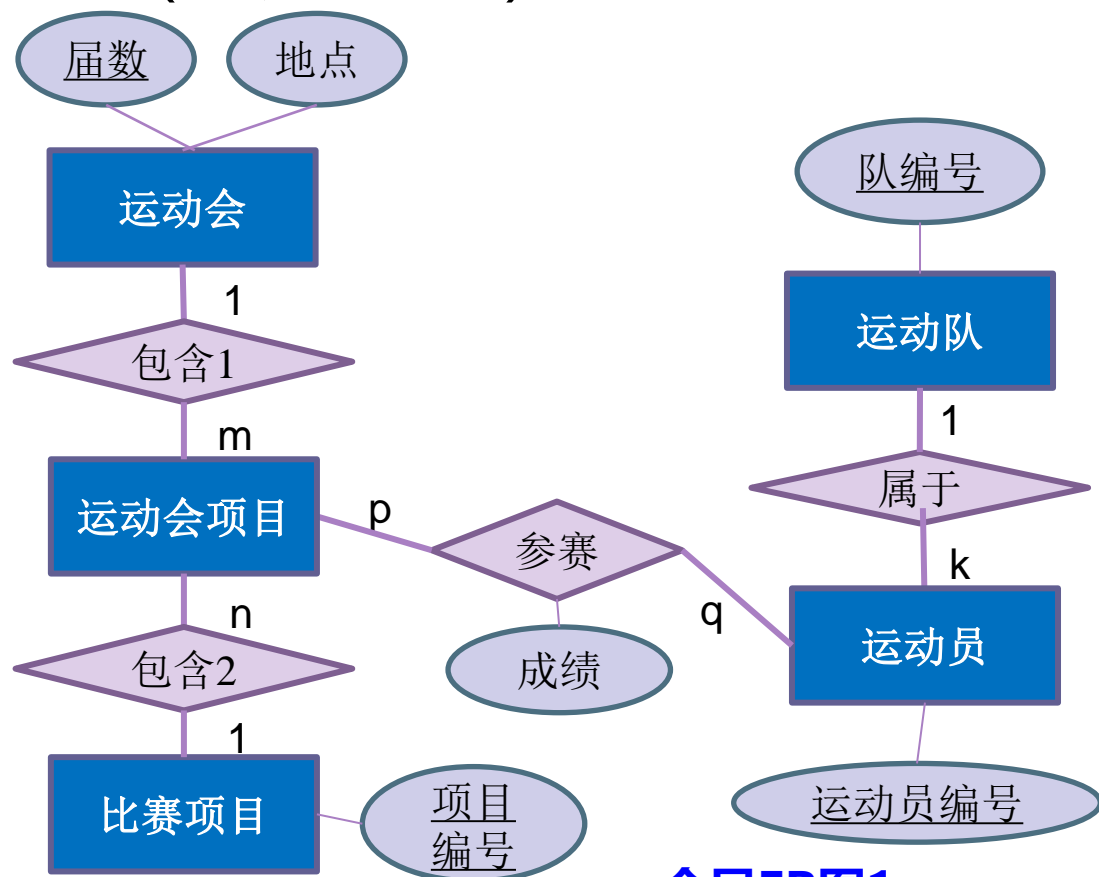
全局ER图 (不是惟一答案)

课堂练习题

3. 对应关系模式。

(建表语句略)

某运动会历经数届，每届在不同地点举行，设有多项体育项目比赛，有若干运动队参加，每个运动队有许多运动员，一个运动员仅可为一个运动队的队员，可参加多届运动会的多个体育项目的比赛。



全局ER图1
(画法1)

运动会 (届数, 运动会名称, 地点)

运动队 (队编号, 队名, ...)

运动员 (运动员编号, 姓名, 队编号...)

比赛项目 (项目编号, 项目名, ...)

运动会项目 (届数, 项目编号, ...)

参赛记录 (届数, 项目编号, 运动员编号, 成绩)

(注: 有颜色的属性是必须表示出来的)

回顾需求:

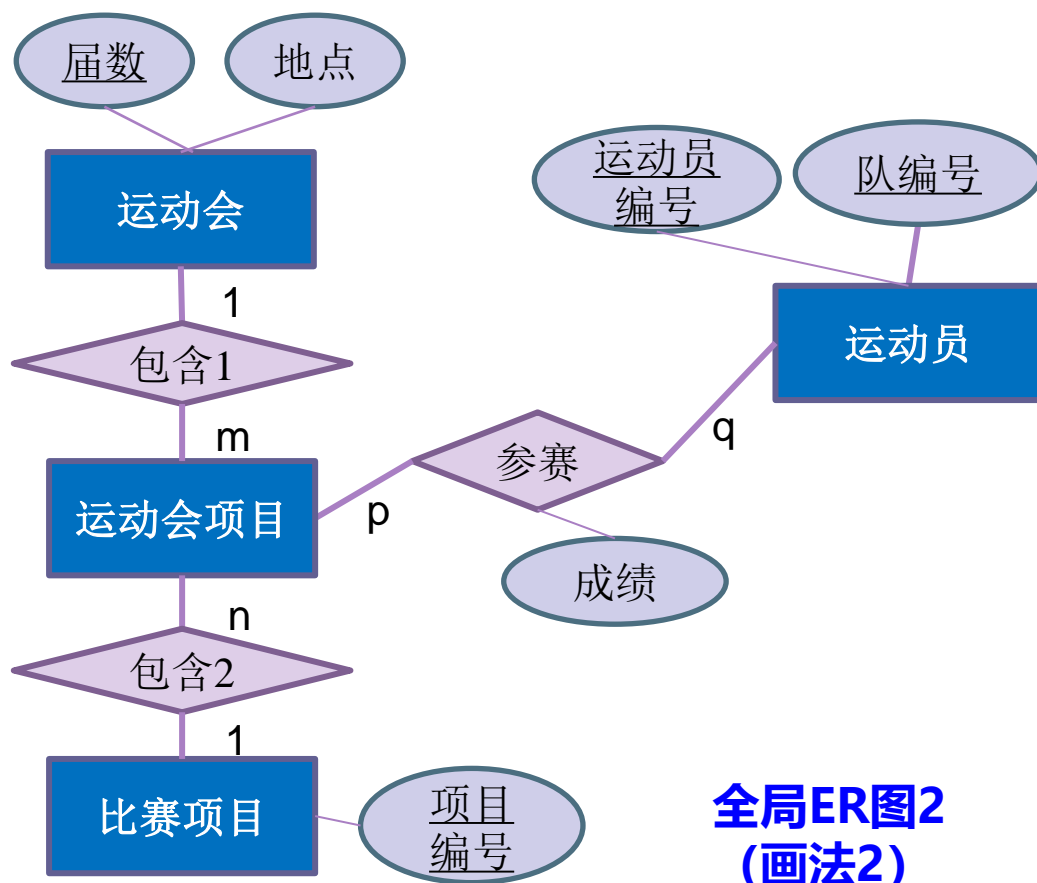
- 1) 求历届运动会参与的运动队和运动员信息。
- 2) 求运动员参加历届运动会的各种比赛项目及成绩。

满足需求!

3. 对应关系模式。

(建表语句略)

某运动会历经数届，每届在不同地点举行，设有多项体育项目比赛，有若干运动队参加，每个运动队有许多运动员，一个运动员仅可为一个运动队的队员，可参加多届运动会的多个体育项目的比赛。



运动会（届数，运动会名称，地点）

运动员 (**运动员编号**, 姓名, **队编号**...)

比赛项目 (项目编号, 项目名, ...)

运动会项目（**届数**，**项目编号**，...）

参赛记录 (届数, 项目编号, 运动员编号, 成绩)

(注：有颜色的属性是必须表示出来的)

回顾需求：

- 1) 求历届运动会参与的运动队和运动员信息。
- 2) 求运动员参加历届运动会的各种比赛项目及成绩。

满足需求!