关系数据理论

单世民

Plan A or Plan B?

Sno	SName	Dname	DLeader	Cno	Grade

Sno	SName	Dname

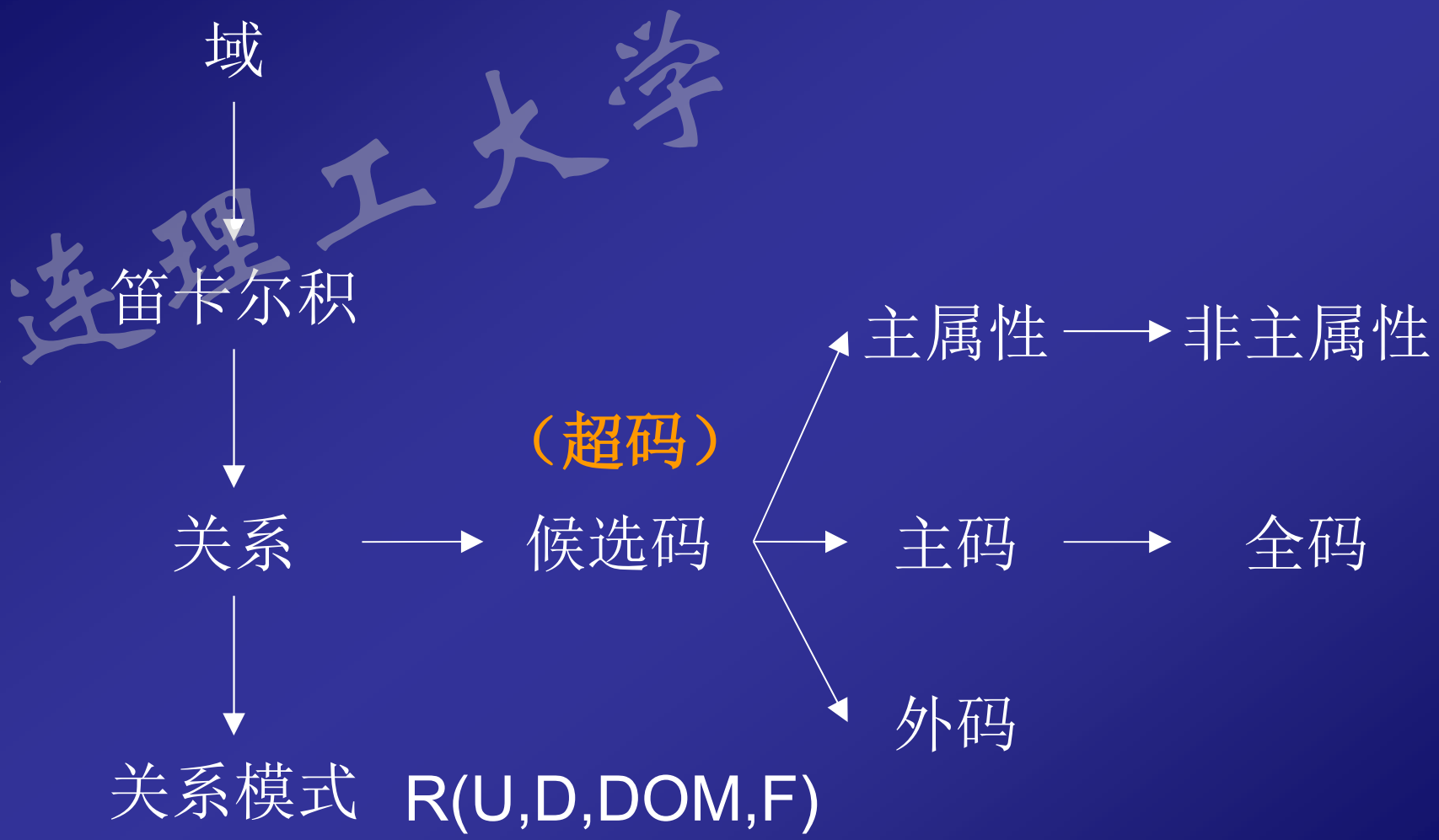
Sno	Cno	Grade

Dname	DLeader

数据模式应规范化

- 在数据库的设计中，如何构造一个适合于它的数据模式，即应该构造几个关系模型，每个关系由哪些属性组成等。这是数据库设计的问题。
- 实际上任何一种数据库应用系统，不论是层次的、网状的还是关系的，都会遇到如何构造合适的数据模型及逻辑结构的问题。
- 规范化理论对于一般的数据库逻辑设计同样具有理论上的意义。

概念回顾



概念回顾

- 关系模式形式化定义
  - **关系模式**：
    对关系的描述称为关系模式。可以形式化的表示为：R（U，D，dom，F）
    其中：
    R：关系名
    U：组成关系的属性名集合
    D：属性来自的域
    dom：属性到域的映射
    F：属性间数据的依赖关系
    简记：
    R（U）,或者R（A1,A2,…An）
  - 关系模式的表示可简化为一个三元组：R（U, F）。当且仅当U上的一个关系r 满足F时，r称为关系模式 R（U, F）的一个关系

数据依赖

- 什么是数据依赖？
  数据依赖是关系中的属性值之间的相互关系。它是现实世界属性间相互联系的抽象，是数据内在的性质，是**语义的体现**。通俗地讲，数据依赖体现了属性之间的对应关系，体现了属性之间的相关性。

- 例：
  在某学校的学生的关系中，只要学号确定了，他（她）的姓名就确定了。这相当于对于一个由学号到姓名的映射 $Sname = f(Sno)$ 这样的数据依赖称为**函数依赖**（Functional Dependency）。记为：Sno->Sname；

数据依赖

- 为什么要关注数据依赖？
  充分了解函数依赖有助于形成好的，规范的关系模式，减少数据冗余。

- 例：
  假设要建立一个学生成绩数据库，要描述的对象包括：学号（Sno）,学生姓名（SName）,系（DName），系负责人（DLeader）,课程号（Cno）和成绩（Grade）。于是可设定其对应的属性集合为：
  U= {Sno,SName,DName,DLeader,Cno,Grade}

数据依赖

- 如果根据以上分析将所有属性放入一个关系模式中（如下例），可否？

Sno	SName	Dname	DLeader	Cno	Cname	Grade

学号(Sno)	姓名(Sname)	系名(Dname)	系主任(DLeader)	课程号(Cno)	课程名(Cname)	成绩(Grade)
1	胡斐	胡氏家族	胡一刀	1	刀法	100
1	胡斐	胡氏家族	胡一刀	2	内功	85
2	段誉	段氏家族	段正南	2	内功	100
3	韦一笑	明教	张无忌	3	轻功	100
4	殷天正	明教	张无忌	2	内功	90
5	乔峰	丐帮	洪七公	2	内功	98
2	段誉	段氏家族	段正南	1	刀法	10
6	裘千丈	铁掌帮	裘千仞	4	歪门邪道	98

数据依赖

- 如果一个系刚好成立但还没有招生，怎样表示存在这个系呢？
- 若插入该系的信息（系名，系负责人），学生信息及选课信息如何表达？
- 若虽然有学生，但还没有成绩，如何表达？

数据依赖

- 若某个系的学生全部毕业了，但系还要保留，这样若删除所有学生的记录，系的信息将不复存在。
- 又若某学生只选了一门课，现在这门课也不选了，删除时会删除学生的信息。
- 又如：每条记录都有系名，系领导的信息，浪费存储，另外换系领导后，所有的记录都要一起更新。

数据依赖

- 问题：怎样才不会发生以上这些问题呢？
  问题的关键源于关系的模式设计有问题，造成了数据冗余，即没有考虑好数据依赖，从而没有设计成好的、规范的模式。

- 由实际中的知识可知，前述情况存在着如下的数据对应关系：
  - 一个系有若干学生，一个学生只属于一个系。
  - 一个系只有一名负责人（正职）。
  - 一个学生可选修多门课程，每门课程有若干学生选修,每个学生选修的每门课程有一个成绩。

数据依赖

- 函数依赖
  函数依赖是最常见的一种数据依赖。

- 两种定义
  - **函数依赖**：设$R(U)$是属性集U上的关系模式，X,Y是U的子集。若对于$R(U)$的任意一个可能的关系r, r中不可能存在两个元组在X上的属性值相等，而在Y上的属性值不等，则称**X函数确定Y**或**Y函数依赖X**。记作 **$X \rightarrow Y$**。X称为这个函数依赖的**决定属性集**(Determinant)。

  - **函数依赖**：r中任意两个元组t,s,若$t[X]=s[X]$，则$t[Y]=s[Y]$，则称$X \rightarrow Y$。

数据依赖

- 对于前例中所述语义，可知存在（但不限于）如下的函数依赖：

  U= {Sno,SName,DName,DLeader,Cno,Grade}

  - 一个系有若干学生，一个学生只属于一个系。

    Sno→DName

  - 一个系只有一名负责人（正职）。

    Dname→DLeader

  - 一个学生可选修多门课程，每门课程有若干学生选修，每个学生选修的每门课程有一个成绩。

    （Sno，Cno）→Grade

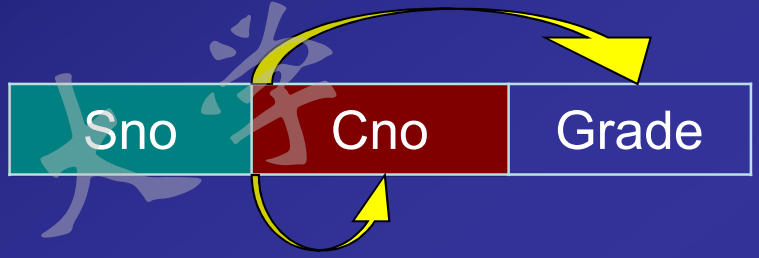
- 若X→Y，并且Y→X，则记为X←→Y。若Y不函数依赖于X，则记为X↛Y。

数据依赖

- 函数依赖-如何理解
  - 函数依赖不是指关系模式*R*的某个或某些关系实例满足的约束条件，而是指**R的所有关系实例均要满足的约束条件**。
  - 函数依赖是语义范畴的概念。**只能根据数据的语义来确定函数依赖**。例如，“姓名→年龄”这个函数依赖只有在不允许有同名人的条件下成立
  - **数据库设计者可以对现实世界作强制的规定**。例如规定不允许同名人出现，函数依赖“姓名→年龄”成立。所插入的元组必须满足规定的函数依赖，若发现有同名人存在，则拒绝装入该元组。

数据依赖

- 函数依赖



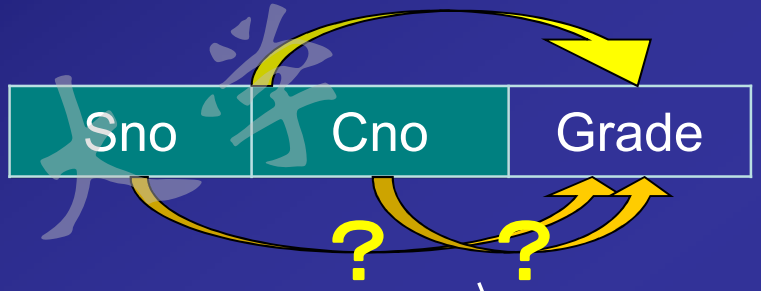
¤ (Sno, Cno) → Grade　　非平凡的函数依赖
¤ (Sno, Cno) → Cno　　平凡的函数依赖

- 形式化定义
  X→Y,但Y ⊆ X则称X→Y是**平凡的函数依赖**。
  X→Y,但Y ⊈ X则称X→Y是**非平凡的函数依赖**。
- 对于任一关系模式，平凡函数依赖都是必然成立，它不反映新的语义。因此若不特别声明，总是讨论非平凡函数依赖。

Ssm

数据依赖

- 函数依赖

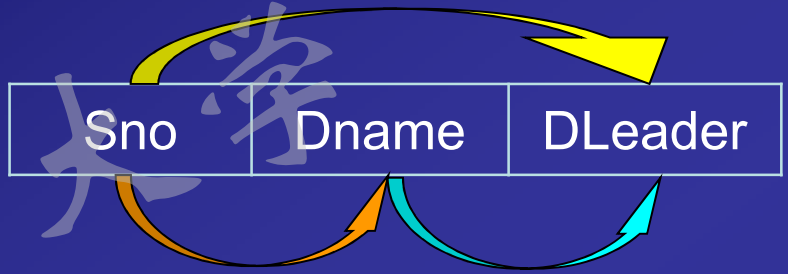


- 由于：Sno $\not\rightarrow$ Grade，Cno $\not\rightarrow$ Grade
  因此：(Sno, Cno) $\xrightarrow{F}$ Grade

- 形式化定义
  在$R(U)$中，如果$X \rightarrow Y$,并且对于X的任何一个真子集X',都有$X' \not\rightarrow Y$,则称Y对X 完全依赖，记作：$X \xrightarrow{F} Y$。
  若$X \rightarrow Y$但Y不完全依赖于X,则称Y对X部分函数依赖，记作：$X \xrightarrow{P} Y$。

数据依赖

- 函数依赖



- Sno → DName，DName → DLeader

- 形式化定义
  在关系模式R(U)中，如果X→Y，Y→Z，且 $Y \subsetneq X$，$Y \not\to X$，则称Z传递函数依赖于X。
  注: 如果Y→X， 即X←→Y，则Z直接依赖于X。

数据依赖

- 码
  在关系模式的所有属性中，各个属性的地位是不一样的。

- 候选码(Candidate key)：
  设K为$R<U,F>$中的属性或属性的组合，若$K \xrightarrow{F} U$，则称K为R的候选码。若候选码多于一个，则选定其中的一个为主码(Primary key)。

数据依赖

- 码
- 包含在任一个候选码中的属性，叫做**主属性**。不包含在任何码中的属性称为**非主属性**。
- 关系模式R中属性或属性组*X*并非*R*的码，但*X*是另外一个关系模式的码，则称*X*是关系*R*的**外码**（Foreign key）。
- 主码和外部码一起提供了表示关系间联系的手段。

规范化

- 规范化理论被用来改造关系模式，通过**分解关系模式**来消除其中不合适的数据依赖，以解决插入异常、删除异常、更新异常和数据冗余问题。

- 1971年Edgar Codd提出了规范化理论，规范化理论研究已经取得了很多的成果。根据**属性数据依赖的情况**，规范化理论采用范式来区分关系的规范化程度（1NF,2NF,3NF, …）。

无损分解

- 规范化过程涉及把一个关系模式分解成几个关系模式，并且这种分解是“可逆”的，这样在分解过程中就不会有信息丢失。规范化过程中的模式分解应该具有**分解的无损性。**

- “没有信息丢失”指的是分解后得到的关系可以经过自然连接得到原来的关系。“可逆”的意思是原来的关系等于它的投影的连接。

无损分解

Sno	DName	Dcount
s2	EE	140
s5	CS	140

Sno	Dcount
s2	140
s5	140

Sno	DName
s2	EE
s5	CS

无损分解

Sno	Dcount
s2	140
s5	140

Dcount	DName
140	EE
140	CS

有损分解

范式

- 范式是符合某一种级别的关系模式的集合。
- 关系数据库中的关系必须满足一定的要求。满足不同程度要求的为不同范式。
- 范式的种类：
  - 第一范式(1NF)
  - 第二范式(2NF)
  - 第三范式(3NF)
  - Boyce-Codd范式(BCNF)
  - 第四范式(4NF)
  - 第五范式(5NF)

范式

- 各种范式之间存在联系：

  1NF ⊃ 2NF ⊃ 3NF ⊃ BCNF ⊃ 4NF ⊃ 5NF

- 某一关系模式R为第n范式，可简记为R∈nNF。

1NF

- 如果关系模式R的所有属性域都是原子（不可分的基本数据项）的，称R属于1NF,记为：R ∈ 1NF。

  注意：在关系模型中就假设所有的属性域都是原子的。即不能出现表中表的情况。但是满足第一范式的关系模式并不一定是一个好的关系模式。

- 这是关系模式的基本要求，后面关系模式的讨论都是满足这一要求的。

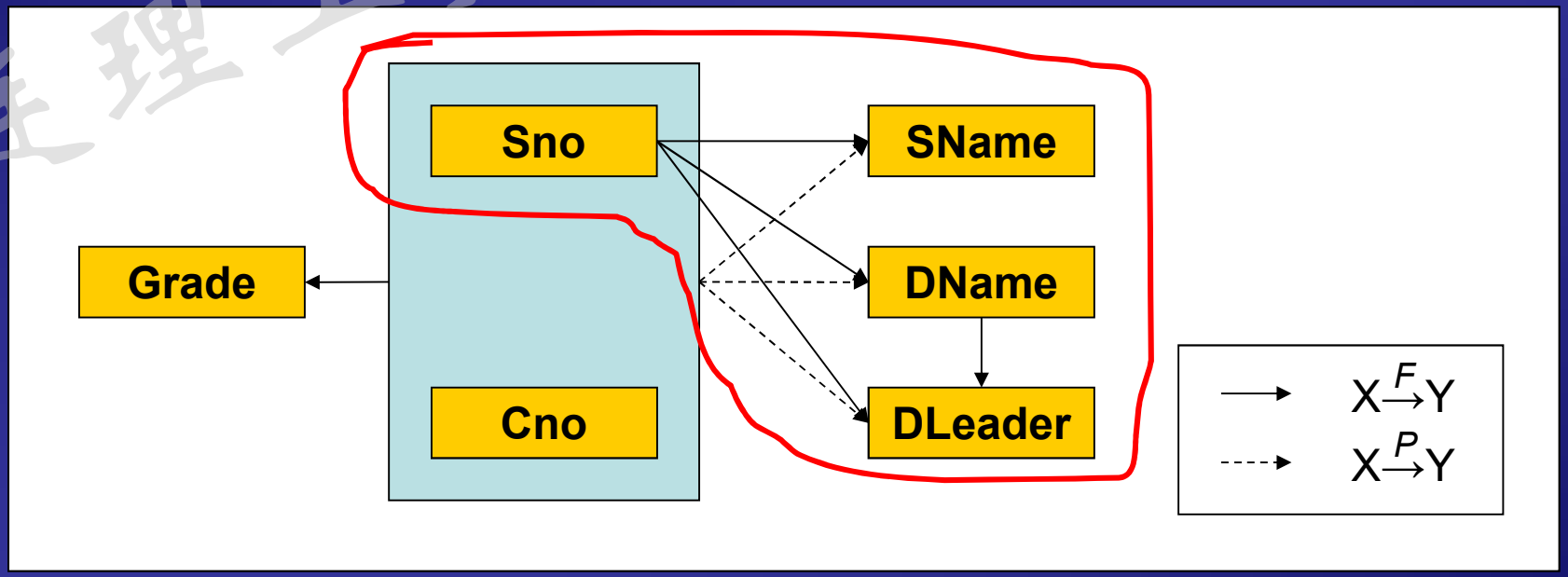
范式示例

Sno	SName	Dname	DLeader	Cno	Grade

- 当前模式（S-D-C）的疑惑：
  - 如果一个系刚好成立但还没有招生，怎样表示存在这个系呢？
  - 若插入该系的信息（系名，系负责人），学生信息如何表达？
  - 若插入该系的信息（系名，系负责人），选课信息如何表达？
  - 若虽然有学生，但还没有成绩，如何表达？
  - 若某个系的学生全部毕业了，但系还要保留，这样若删除所有学生的记录，系的信息将不复存在。如何避免？
  - 又若某学生只选了一门课，现在这门课也不选了，删除时会删除学生的信息。如何避免？
  - 每条记录（针对每个学生的每条选课信息）都有系名，系领导的信息，浪费存储。
  - 另外换系领导后，所有的记录都要一起更新。如何避免？

范式示例

- 关系模式S-D-C中的函数依赖



某些属性所描述的不是由主码所确定的实体的信息
应把逻辑上独立的信息放在独立的关系中。

范式示例

Sno	SName	Dname	DLeader

Sno	Cno	Grade

- 相比于S-D-C，当前模式（S-D及S-C）具有如下优点：
  - 若虽然有学生，但还没有成绩，可以直接将学生信息加入到S-D模式的实例中
  - 若某学生只选了一门课，现在这门课也不选了，删除时只删除S-C模式的实例中的元组，并不会删除学生的信息。
  - 对于某个学生的每个课程成绩记录，不必重复存储有关系名和系领导的信息，减少了冗余。
  - 另外，降低了更新时的复杂性。
  - 插入系的信息时不必考虑如何表达选课信息。

2NF

- 模式（S-D及S-C）属于2NF

- 定义：
  若关系模式R∈1NF，并且每一个非主属性都完全函数依赖于R的码，则R∈2NF。

- S-C（Sno, Cno, Grade）∈2NF
  S-D（Sno,SName,DName, DLeader）∈2NF

范式示例

Sno	SName	Dname	DLeader	Cno	Grade

- 当前模式（S-D-C）的疑惑：
  - 如果一个系刚好成立但还没有招生，怎样表示存在这个系呢？
  - 若插入该系的信息（系名，系负责人），学生信息如何表达？
  - **若插入该系的信息（系名，系负责人），选课信息如何表达？**
  - **若虽然有学生，但还没有成绩，如何表达？**
  - 若某个系的学生全部毕业了，但系还要保留，这样若删除所有学生的记录，系的信息将不复存在。如何避免？
  - **又若某学生只选了一门课，现在这门课也不选了，删除时会删除学生的信息。如何避免？**
  - **每条记录（针对每个学生的每条选课信息）都有系名，系领导的信息，浪费存储。**
  - **另外换系领导后，所有的记录都要一起更新。如何避免？**

范式示例

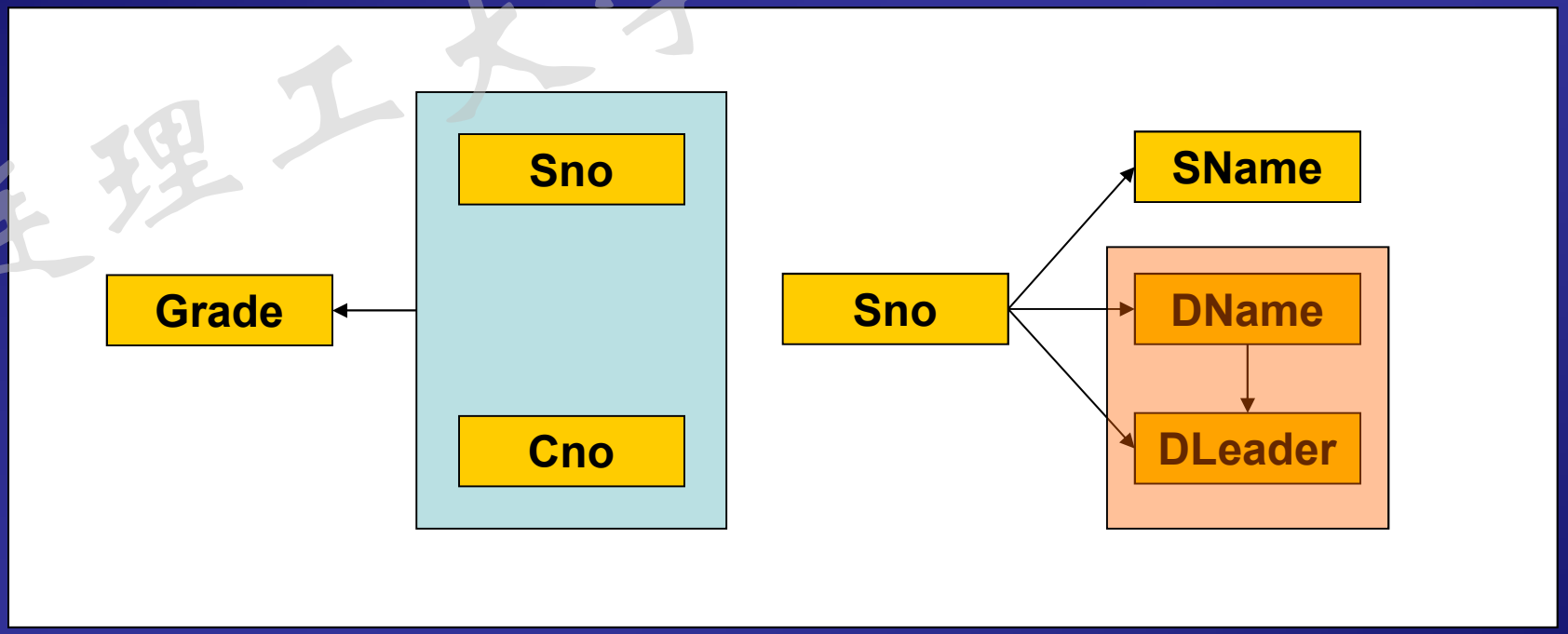
Sno	SName	Dname	DLeader

Sno	Cno	Grade

- 但是，对模式（S-D，S-C）仍然存在疑惑：
  - 如果一个系刚好成立但还没有招生，怎样表示存在这个系呢？
  - 若插入该系的信息（系名，系负责人），学生信息如何表达？
  - 若某个系的学生全部毕业了，但系还要保留，这样若删除所有学生的记录，系的信息将不复存在。如何避免？
  - 每条学生记录都有系名，系领导的信息，浪费存储
  - 另外换系领导后，对所有学生的记录都要一起更新。如何避免？

范式示例

- 关系模式S-D，S-C中的函数依赖



某些属性所描述的不是由主码所确定的实体的信息
应把逻辑上独立的信息放在独立的关系中。

范式示例

Sno	SName	Dname

Dname	DLeader

Sno	Cno	Grade

- 相比于S-D，S-C，当前模式（S、D及S-C）具有如下优点：
  - 如果一个系刚好成立但还没有招生，那么可以将系的信息放入模式D的实例中，而不必理会此系中有无学生。
  - 若插入该系的信息（系名，系负责人），不必理会学生信息如何表达
  - 若某个系的学生全部毕业了，但系还要保留，这样可以删除模式S的实例中所有学生的记录，但模式D的实例中系的信息仍将存在。
  - 避免了对每条学生记录都存储系名，系领导的信息，进一步减少了数据冗余。
  - 另外，换系领导后，不必对所有学生的记录都要更新，减小了更新的复杂度。

3NF

- 模式S、D及S-C属于3NF

- 定义：
  关系模式$R<U, F>$ 中若不存在这样的码X、属性组Y及非主属性Z（$Z \subsetneq Y$），使得$X \rightarrow Y$，$Y \not\rightarrow X$，$Y \rightarrow Z$成立，则称$R<U, F> \in$3NF。

- S（Sno，SName，DName）∈3NF
  D（DName，DLeader）∈ 3NF
  S-C（Sno，Cno，Grade）∈ 3NF

3NF

- 若R∈3NF，则R的每一个非主属性**既不部分函数依赖于候选码也不传递函数依赖于候选码**。
- 如果R∈3NF，则R也是2NF。

- 将一个2NF的关系分解为多个3NF的关系，可在一定程度上解决原关系中存在的插入异常、删除异常、数据冗余度大、修改复杂等问题。但是，这并不能完全消除关系模式中的各种异常情况和数据冗余。

保持函数依赖

- 思考
- {Sno,Sname,Dname,Dleader}
- 可以分解成
  - {Sno,Sname,Dname},{Dname,Dleader}
- 还可以分解成什么？
  - 还可以分解成
    {Sno,Sname,Dname},{Sno,Dleader}
  - 这样分解“好不好”？

保持函数依赖

- 分解方案
  {Sno,Sname,Dname},{Sno,Dleader}
  并不好
- 原因在于原有的函数依赖DName→DLeader变成了跨越两个关系变量的数据库级约束条件。两个投影不是相互独立的。为保证不违反函数依赖DName→DLeader，更新其中任何一个投影时都必须对两个投影实行监控。

- 考虑学生转系的情况。

保持函数依赖

- 投影独立性概念为从可能存在的多种分解方法中选择一个合适的分解提供了一个准则。
- 具体地说，一个投影相互独立的分解比相互不独立的分解“好”。这种思想被称为“保持函数依赖”。
- 当且仅当满足下列条件时，关系R的两个投影R1和R2相互独立：
  - R中的所有函数依赖都是R1和R2的函数依赖的逻辑推论
  - R1和R2的相同属性至少组成它们之中一个的候选码

BCNF

- 3NF没有很好地处理一般情况下的问题，更确切的说，是没有很好地处理具有下列特性的关系：
  - 具有一个或多个候选码
  - 候选码是复合的
  - 候选码之间是重叠的（即至少由一个属性是相同的）

BCNF

- 假设学生姓名SName是唯一的，那么下面的关系模式是否满足3NF？

  {Sno,SName,Cno,Grade}

- 存不存在“问题”？

- 存在问题，存在数据冗余，也会产生更新异常。

- 为解决问题，可以将其分解为

  {Sno,SName} {Sno,Cno,Grade}

  或

  {Sno,SName} {SName,Cno,Grade}

BCNF

- 设关系模式R<U，F>∈1NF，如果对于R的每个函数依赖X→Y，若Y不属于X，则X必含有候选码，那么R∈BCNF。

- 当且仅当一个关系变量的所有非平凡的、左部不可约的函数依赖的决定因素是候选码，则该关系变量属于BCNF

- 若R∈BCNF
  - 每一个决定属性集（因素）都包含（候选）码
  - R中的所有属性（主，非主属性）都完全函数依赖于码
  - R∈3NF（证明）
  - 若R∈3NF 则R不一定∈BCNF
  - 如果R∈3NF，且R只有一个候选码，则R必属于BCNF。

BCNF

- 所有非主属性都完全函数依赖于每个候选码
- 所有主属性都完全函数依赖于每个不包含它的候选码
- 没有任何属性完全函数依赖于非码的任何一组属性

BCNF

- 关系变量R无损分解至一系列属于BCNF的投影集D的算法
  1. 初始化D为只包含R
  2. 对D中的每个非BCNF关系变量T，执行步骤3和4
  3. 设X→Y为T的不满足BCNF要求的函数依赖。
  4. 用T的两个投影（在X和Y上的投影及在除Y中属性之外的所有属性上的投影）代替D中的T

范式示例

- 假设仓库管理关系表为StorehouseManage(仓库ID, 配件ID, 管理员ID, 数量)。一个管理员只在一个仓库工作；一个仓库有多个管理员；每个仓库里一种型号的配件由专人负责，但一个人可以管理几种配件；同一种型号的配件可以分放在几个仓库中。

WID	PID	EID	QNT

- 这个数据库表中存在如下决定关系：
  - **(WID, PID)** →QNT
  - **(WID, PID)** → EID
  - EID →WID
  - **(EID,PID)** →QNT

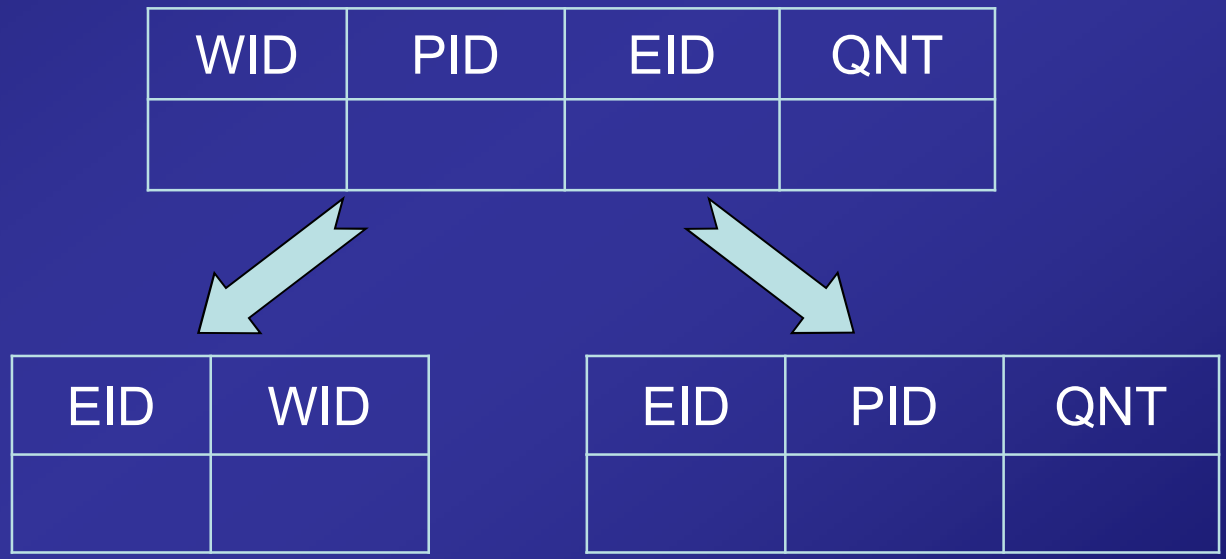
范式示例

- (仓库ID, 配件ID)和(管理员ID, 配件ID)都是StorehouseManage的候选码，表中唯一的非主属性为数量(QNT)，它是符合第三范式的。

- 当前模式的疑惑：
  - 当仓库被清空后，所有“配件ID”和“数量”信息被删除的同时，“仓库ID”和“管理员ID”信息也被删除了。
  - 当仓库没有存储任何物品时，无法给仓库分配管理员。
  - 如果仓库换了管理员，则表中所有行的管理员ID都要修改。

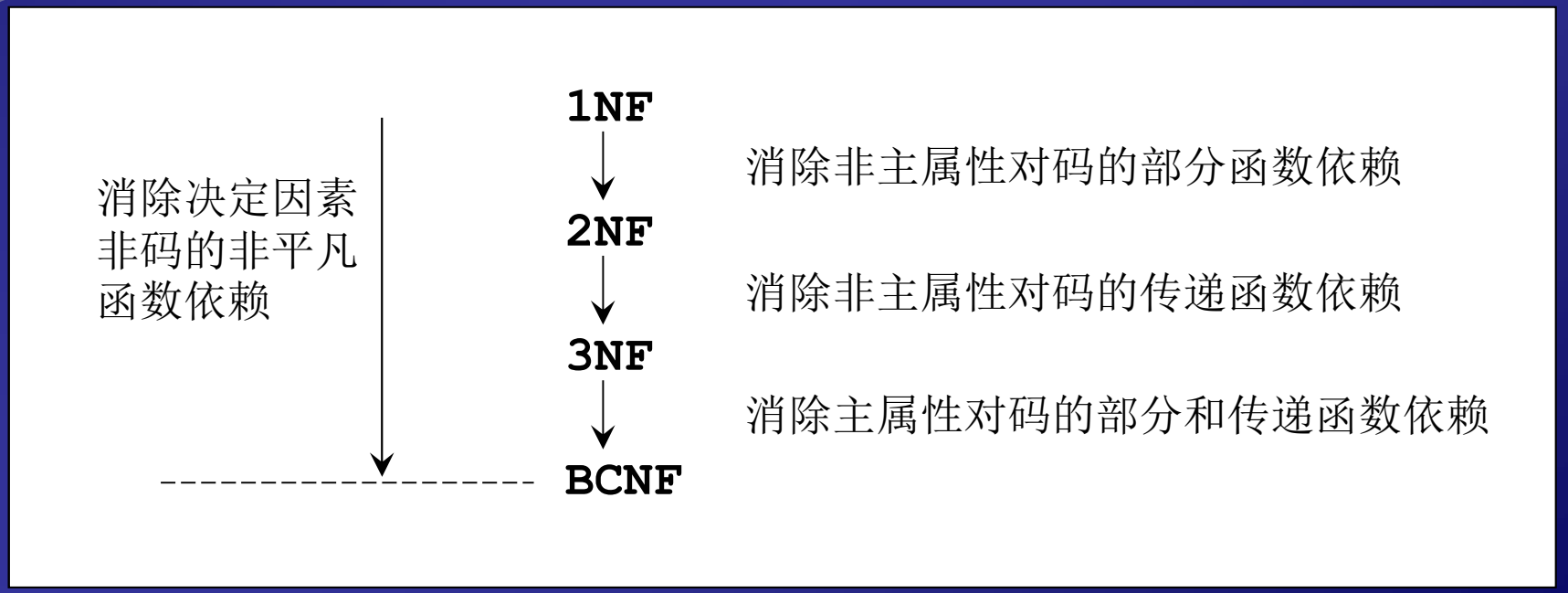
范式示例

- 当前模式存在如下数据依赖关系：
  - EID → WID（存在部分函数依赖）
  - 分解后丢失了数据依赖(WID, PID) → EID



规范化

- 规范化的思想是逐步消除数据依赖中不合适的部分，使模式中的各关系模式达到某种程度的“分离”。即用“一事一地”的模式设计原则，让一个关系描述一个概念、一个实体或实体间的一种联系。若多于一个概念就把它“分离”出去。规范化实质上是概念的单



练习

- 设要建立一个车辆驾驶记录管理数据库，设一辆车可由多个驾驶员驾驶，每个驾驶员也可以驾驶多辆车，需要记录驾驶日期，使用时长；每个驾驶员可以有多个驾驶证，但每个驾驶证只能供一个驾驶员使用。描述车辆的属性有：车牌号码，名称；描述驾驶员的属性有：驾驶证号，发证单位。其中 车牌号码是车辆的标识属性，身份证号是驾驶员的标识属性，驾驶证号是驾驶证的标识属性。

- 假设该数据库系统的关系模式如下：
- R（车牌号码，名称，身份证号，姓名，驾驶证号，发证单位，驾驶日期，使用时长）
- 请给出该关系模式的函数依赖集和候选关键字。
- 请将上题中的关系模式R分解成满足3NF的关系模式的集合，并说明理由。

小结

- 规范化
  - 函数依赖
  - 码
  - 范式
  - 2NF
  - 3NF
  - BCNF

数据的联系

规范化

- 关系数据库的规范化理论是数据库逻辑设计的工具。
- 一个关系只要其分量都是不可分的数据项，它就是规范化的关系，但这只是最基本的规范化。
- 规范化程度可以有多个不同的级别
- 规范化程度过低的关系不一定能够很好地描述现实世界，可能会存在插入异常、删除异常、修改复杂、数据冗余等问题

规范化

- 一个低一级范式的关系模式，通过模式分解可以转换为若干个高一级范式的关系模式集合，这种过程就叫关系模式的规范化
- 不能说规范化程度越高的关系模式就越好，在设计数据库模式结构时，必须对现实世界的实际情况和用户应用需求作进一步分析，确定一个合适的、能够反映现实世界的模式
- 上面的规范化步骤可以在其中任何一步终止

大连理工大学

The End

Ssm

大连理工大学