

数据库设计

Email: 18442056@QQ.com

课程内容回顾

- 数据库有哪些基本操作？
 - 建库
 - 建表
 - 加约束
 - 创建登录帐户
- 基本的数据操纵语句有哪些？语法是？
 - 增（INSERT）
 - 删（DELETE）
 - 改（UPDATE）
 - 查（SELECT）
- 常用的聚合函数有哪些？
- 表连接分为哪几种类型？

课程目标

- 了解设计数据库的基本步骤
- 熟练使用T-SQL实现建库、建表、加约束
- 掌握T-SQL编程，实现功能强大的查询
- 掌握创建索引、视图，快速访问数据库
- 掌握创建存储过程，实现复杂的业务规则

学习目标

- 了解设计数据库的步骤
- 掌握如何绘制数据库的E-R图
- 理解数据库的规范化—三大范式

为什么需要设计数据库



修建茅屋需要设计吗？



修建大厦需要设计吗？

结论：当数据库比较复杂时我们需要设计数据库

为什么需要设计数据库

- 良好的数据库设计
 - 节省数据的存储空间
 - 能够保证数据的完整性
 - 方便进行数据库应用系统的开发



- 糟糕的数据库设计：
 - 数据冗余、存储空间浪费
 - 内存空间浪费
 - 数据更新和插入的异常

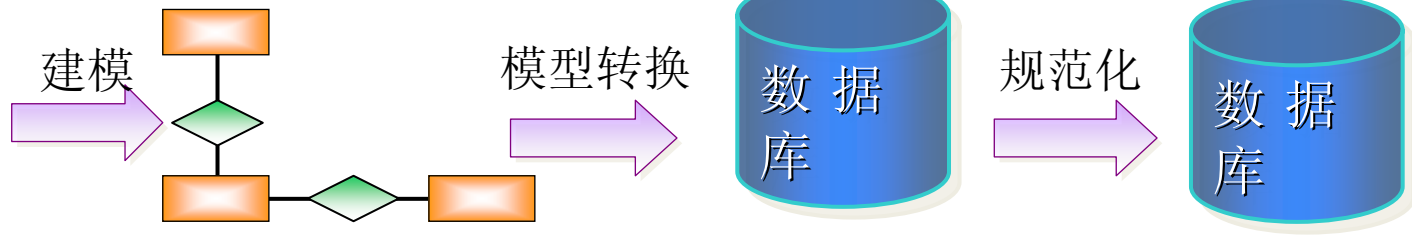


软件项目开发周期

现实世界

信息世界

数据库世界



- 需求分析阶段：分析客户的业务和数据处理需求；
- 概要设计阶段：设计数据库的E-R模型图，确认需求信息的正确和完整；
- 详细设计阶段：将E-R图转换为多张表，进行逻辑设计，并应用数据库设计的三大范式进行审核；
- 代码编写阶段：选择具体数据库进行物理实现，并编写代码实现前端应用；
- 软件测试阶段：
- 安装部署：

设计数据库的步骤

- 收集信息：

与该系统有关人员进行交流、坐谈，充分理解数据库需要完成的任务

BBS论坛的基本功能：

用户注册和登录，后台数据库需要存放用户的注册信息和在线状态信息；

用户发贴，后台数据库需要存放贴子相关信息，如贴子内容、标题等；

论坛版块管理：后台数据库需要存放各个版块信息，如版主、版块名称、贴子数等；

设计数据库的步骤

- 标识对象（实体—Entity）

标识数据库要管理的关键对象或实体

实体一般是名词：

用户：论坛普通用户、各板块的版主。

用户发的主贴

用户发的跟贴（回帖）

版块：论坛的各个版块信息

设计数据库的步骤

标识每个实体的属性（Attribute）

论坛用户
昵称
密码
电子邮件
生日
性别
用户的等级
备注信息
注册日期
状态
积分

主贴
发贴人
发贴表情
回复数量
标题
正文
发贴时间
点击数
状态
最后回复时间

回帖
贴子编号
回帖人,
回帖表情
标题
正文
回帖时间
点击数

版块
版块名称
版主
本版格言
点击率
发贴数

设计数据库的步骤

- 标识对象之间的关系（Relationship）

跟贴和主贴有主从关系：我们需要在跟贴对象中表明它是谁的跟贴；


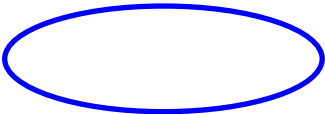
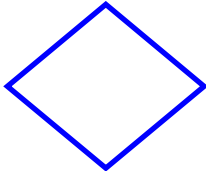
版块和用户有关系：从用户对象中可以根据版块对象查出对应的版主用户的情况；

主贴和版块有主从关系：需要表明发贴是属于哪个版块的；

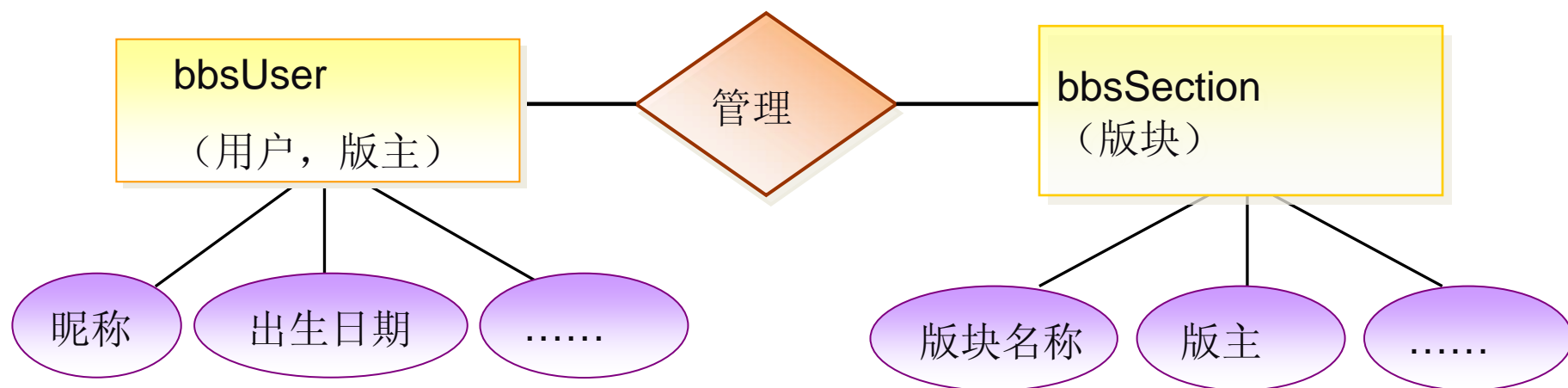
跟贴和版块有主从关系：需要表明跟贴是属于哪个版块的；

绘制E-R图

- E-R（Entity—Relationship）实体关系图

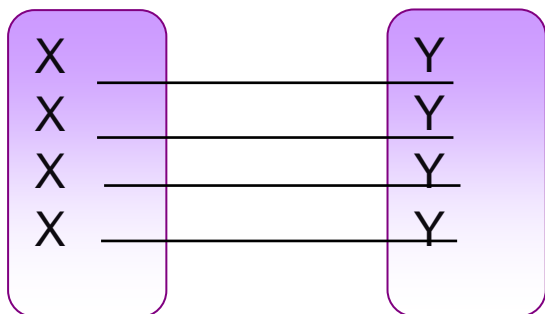
符合	含义
	实体，一般是名词
	属性，一般是名词
	关系，一般是动词

绘制E-R图

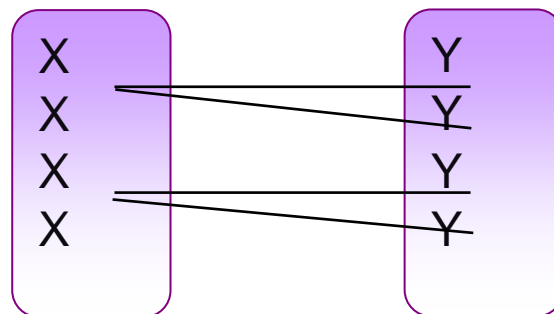


绘制E-R图

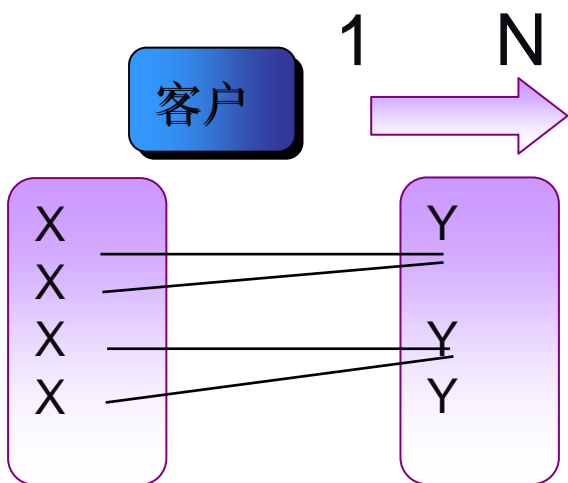
- 映射基数



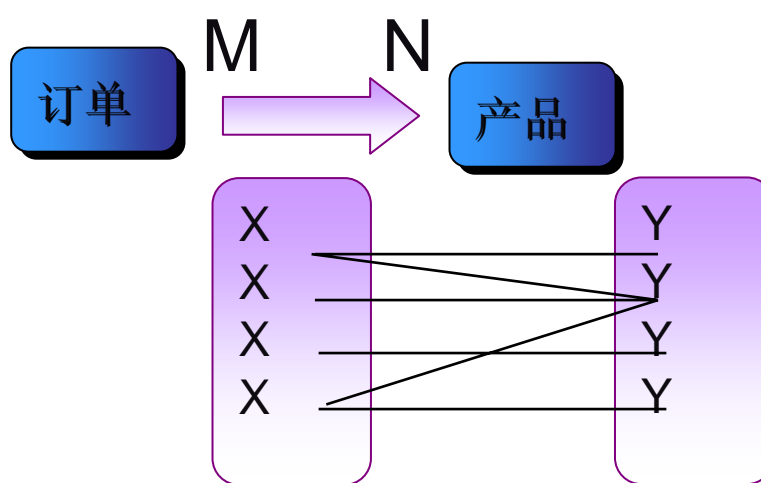
一对一



一对多

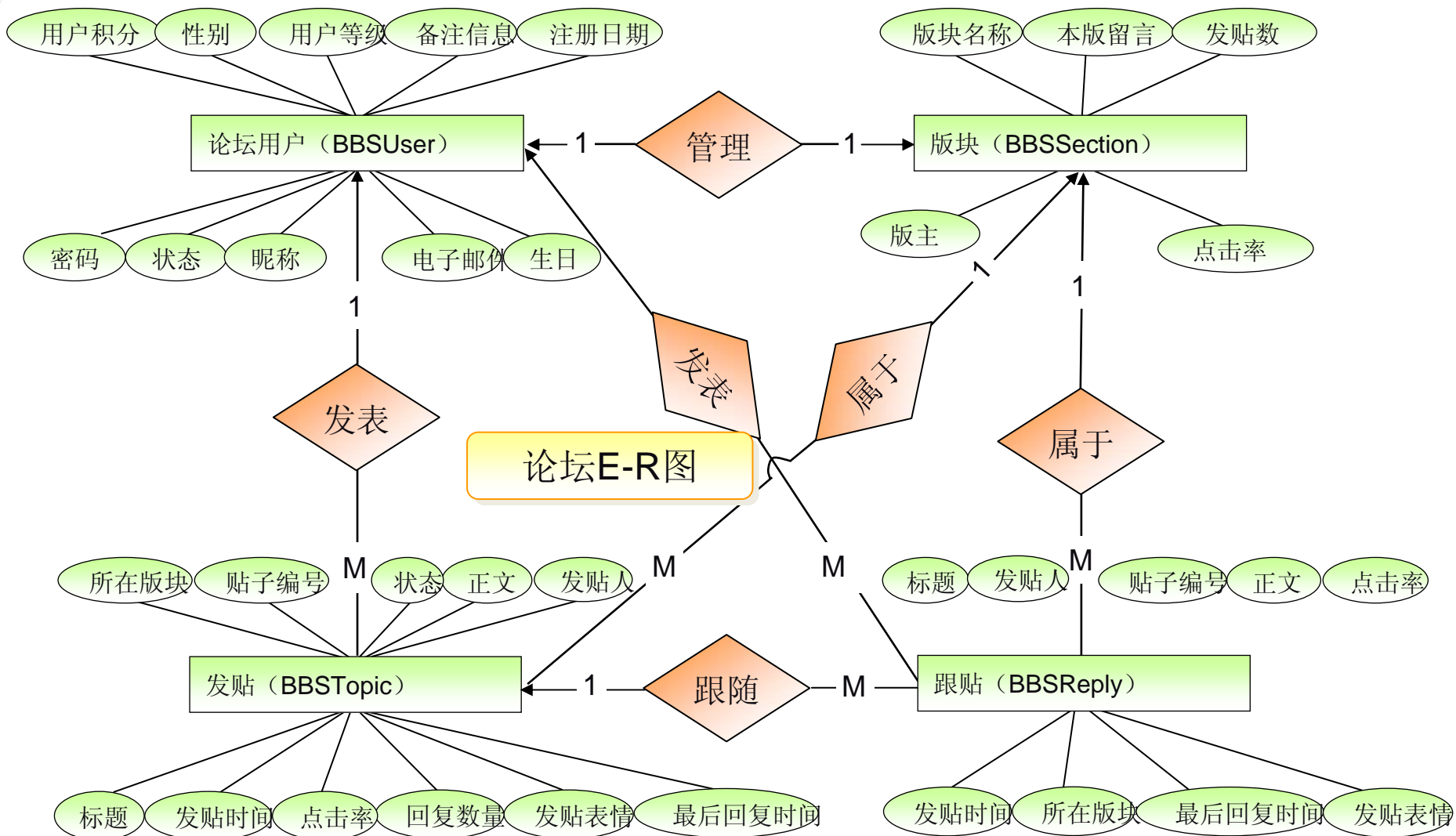


多对一



多对多

绘制E-R图



如何将E-R图转换为表

- 将各实体转换为对应的表，将各属性转换为各表对应的列
- 标识每个表的主键列，需要注意的是：没有主键的表添加ID编号列，它没有实际含义，用于做主键或外键，例如用户表中的“UID”列，版块表中添加“SID”列，发贴表和跟贴表中的“TID”列
- 在表之间建立主外键，体现实体之间的映射关系

如何将E-R图转换为表

UID主键

BBSUser（论坛用户）表

UID（用户编号，主键）

UName（用户昵称）

UPassword（密码）

UEmail（电子邮件）

UBirthday（生日）

USex（性别）

UClass（用户等级）

UStatement（用户备注）

URegDate（注册日期）

UState（用户状态）

UPoint（用户积分）

TID主键

BBSTopic（发帖）表

TID（标识主键列）

TNumber（帖子编号）

TSID（所在版块）

TUID（发帖人）

TReplyCount（回复数）

TEmotion（回复表情）

TTopic（主题）

TContents（正文）

TTime（回复时间）

TClickCount（点击数）

TFlag（状态）

TLastClickT（最后回复时间）

RID主键

BBSReply（跟贴）表

RID（标识主键列）

RNumber（帖子编号）

RTID（回复的主贴）

RSID（所在版块编号）

RUID（发帖人编号）

REmotion（发帖表情）

RTopic（主题）

RContents（正文）

RTime（发帖时间）

SID主键

BBSSection（版块）表

SID（版块编号，主键）

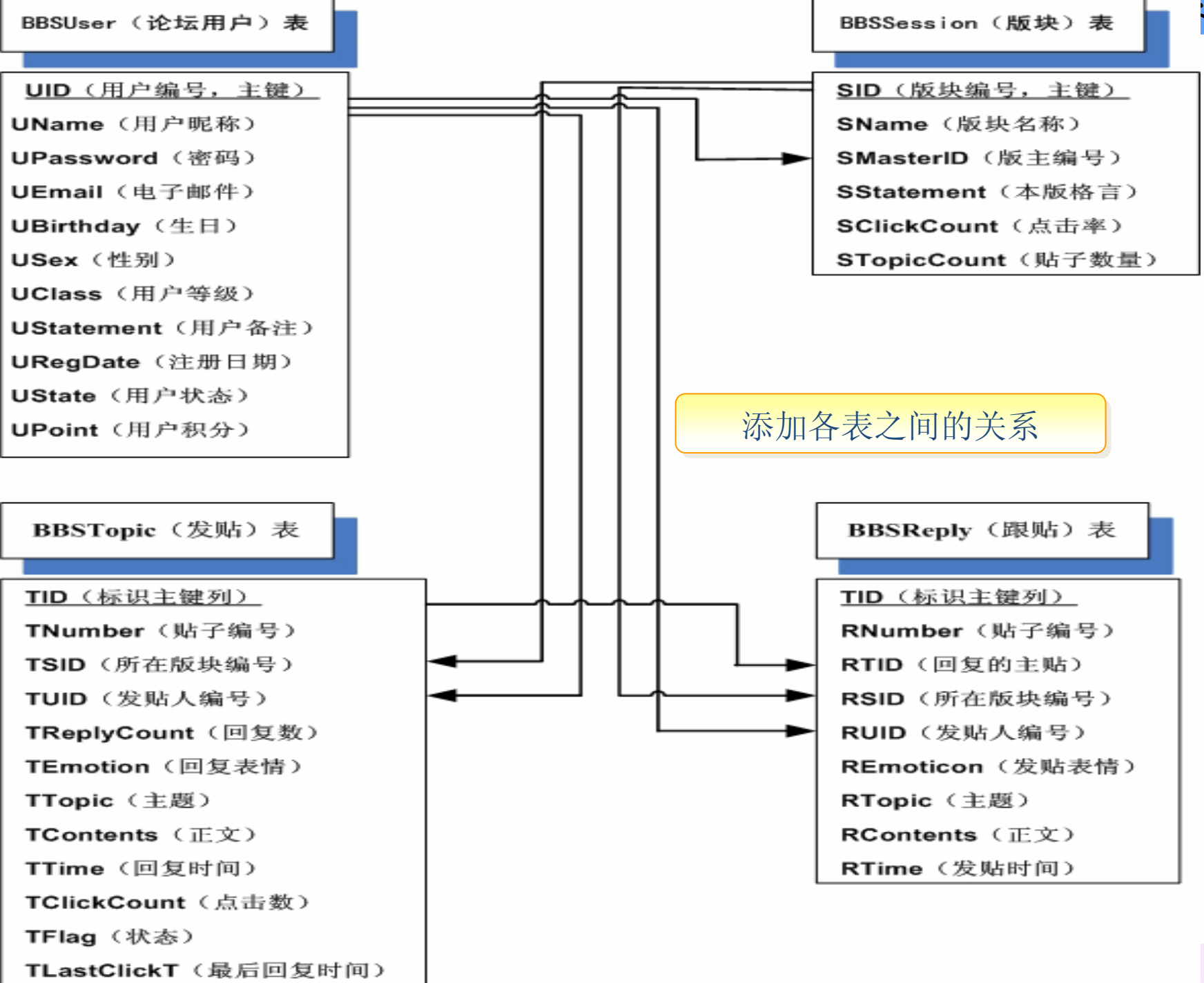
SName（版块名称）

SMasterID（版主编号）

SStatement（本版格言）

SClickCount（点击率）

STopicCount（帖子数量）

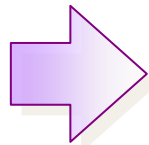


数据规范化

- 仅有好的RDBMS并不足以避免数据冗余，必须在数据库的设计中创建好的表结构
- Dr E.F.codd 最初定义了规范化的三个级别，范式是具有最小冗余的表结构。这些范式是：
 - 第一范式(1st NF — First Normal Fromate)
 - 第二范式(2nd NF—Second Normal Fromate)
 - 第三范式(3rd NF— Third Normal Fromate)

第一范式 (1st NF)

BuyerID	Address
1	中国北京市
2	美国纽约市
3	英国利物浦
4	日本东京市
...	...



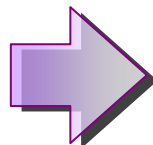
BuyerID	Country	City
1	中国	北京
1	中国	北京
4	日本	东京
2	美国	纽约
...

- 第一范式的目标是确保每列的原子性
- 如果每列都是不可再分的最小数据单元（也称为最小的原子单元），则满足第一范式（1NF）

第二范式 (2nd NF)

Orders

字 段	例 子
订单编号	001
产品编号	A001
订购日期	2000-2-3
价 格	\$29.00
...	...



Orders

字 段	例 子
订单编号	001
订购日期	2000-2-3

Products

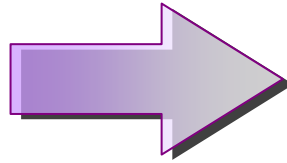
字 段	例 子
产品编号	A001
价 格	\$29.00

- 如果一个关系满足1NF，并且除了主键以外的其他列，都依赖与该主键，则满足第二范式（2NF）
- 第二范式要求每个表只描述一件事情

第三范式 (3rd NF)

Orders

字 段	例 子
订单编号	001
订购日期	2000-2-3
顾客编号	AB001
顾客姓名	Tony
...	...



Orders

字 段	例 子
订单编号	001
订购日期	2000-2-3
顾客编号	AB001
...	...

- 如果一个关系满足2NF，并且除了主键以外的其他列都不传递依赖于主键列，则满足第三范式（3NF）

规范化实例

假设某建筑公司要设计一个数据库。公司的业务规

则概括说明如下：

- 公司承担多个工程项目，每一项工程有：工程号、工程名称、施工人员等
- 公司有多名职工，每一名职工有：职工号、姓名、性别、职务（工程师、技术员）等
- 公司按照工时和小时工资率支付工资，小时工资率由职工的职务决定（例如，技术员的

规范化实例

工程号	工程名称	职工号	姓名	职务	小时工资率	工时	实发工资
A1	花园大厦	1001	齐光明	工程师	65	13	845.00
		1002	李思岐	技术员	60	16	960.00
		1004	葛宇宏	律师	60	19	1140.00
			小计				2945.00
A2	立交桥	1001	齐光明	工程师	65	15	975.00
		1003	鞠明亮	工人	55	17	935.00
			小计				1910.00
A3	临江饭店	1002	李思岐	技术员	60	18	1080.00
		1004	葛宇洪	技术员	60	14	840.00
			小计				1920.00

图-1 某公司的工资表

规范化实例

工程号	工程名称	职工号	姓名	职务	小时工资率	工时
A1	花园大厦	1001	齐光明	工程师	65	13
A1	花园大厦	1002	李思岐	技术员	60	16
A1	花园大厦	1001	齐光明	工程师	65	13
A1	花园大厦	1003	鞠明亮	工人	55	17
A3	临江饭店	1002	李思岐	技术员	60	18
A3	临江饭店	1004	葛宇洪	技术员	60	14

图-2 某公司的项目工时表

规范化实例

- 表中包含大量的冗余，可能会导致数据异常：

- 更新异常

例如，修改职工号=1001的职务，则必须修改所有职工号=1001的行

- 添加异常

若要增加一个新的职工时，首先必须给这名职工分配一个工程。或者为了添加一名新职工的数据，先给这名职工分配一个虚拟的工程。（因为主关键字不能为空）

- 删除异常

例如，1001号职工要辞职，则必须删除所有职工号=1001的数据行。这样的删除操作，很可能丢失了其它有用的数据

规范化实例

- 采用这种方法设计表的结构，虽然很容易产生工资报表，但是每当一名职工分配一个工程时，都要重复输入大量的数据。这种重复的输入操作，很可能导致数据的不一致性。

应用范式规范化设计

一张表描述的多件事情，如图-3所示。

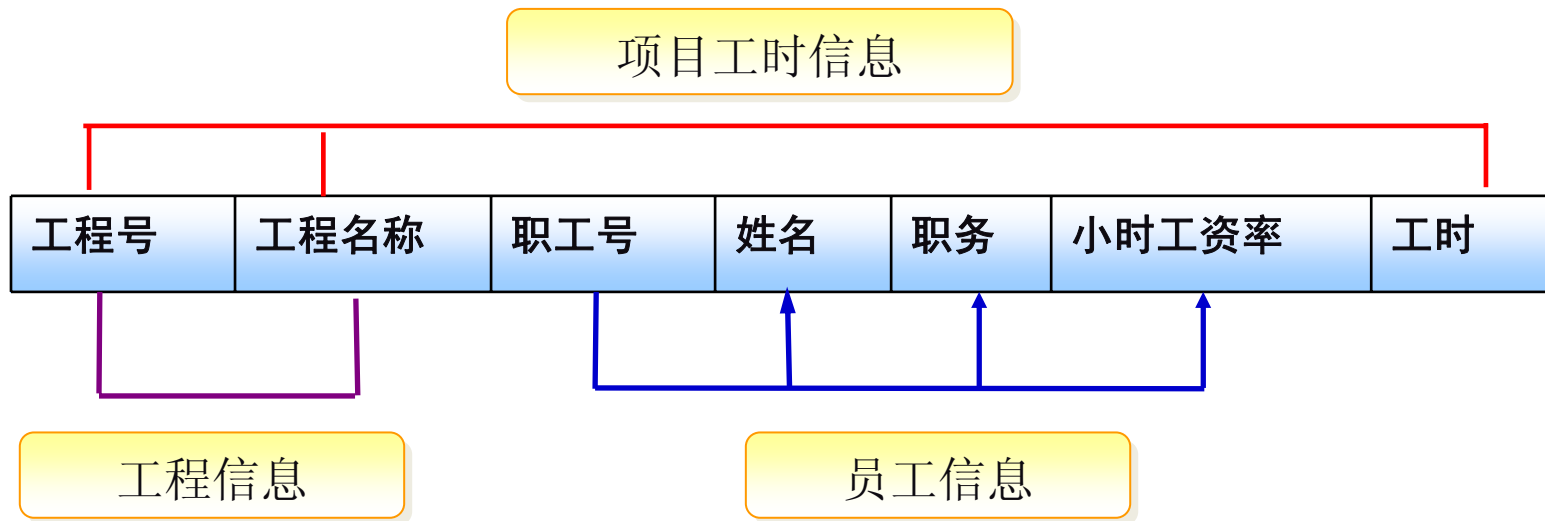


图-3 函数依赖图

应用第二范式规范化

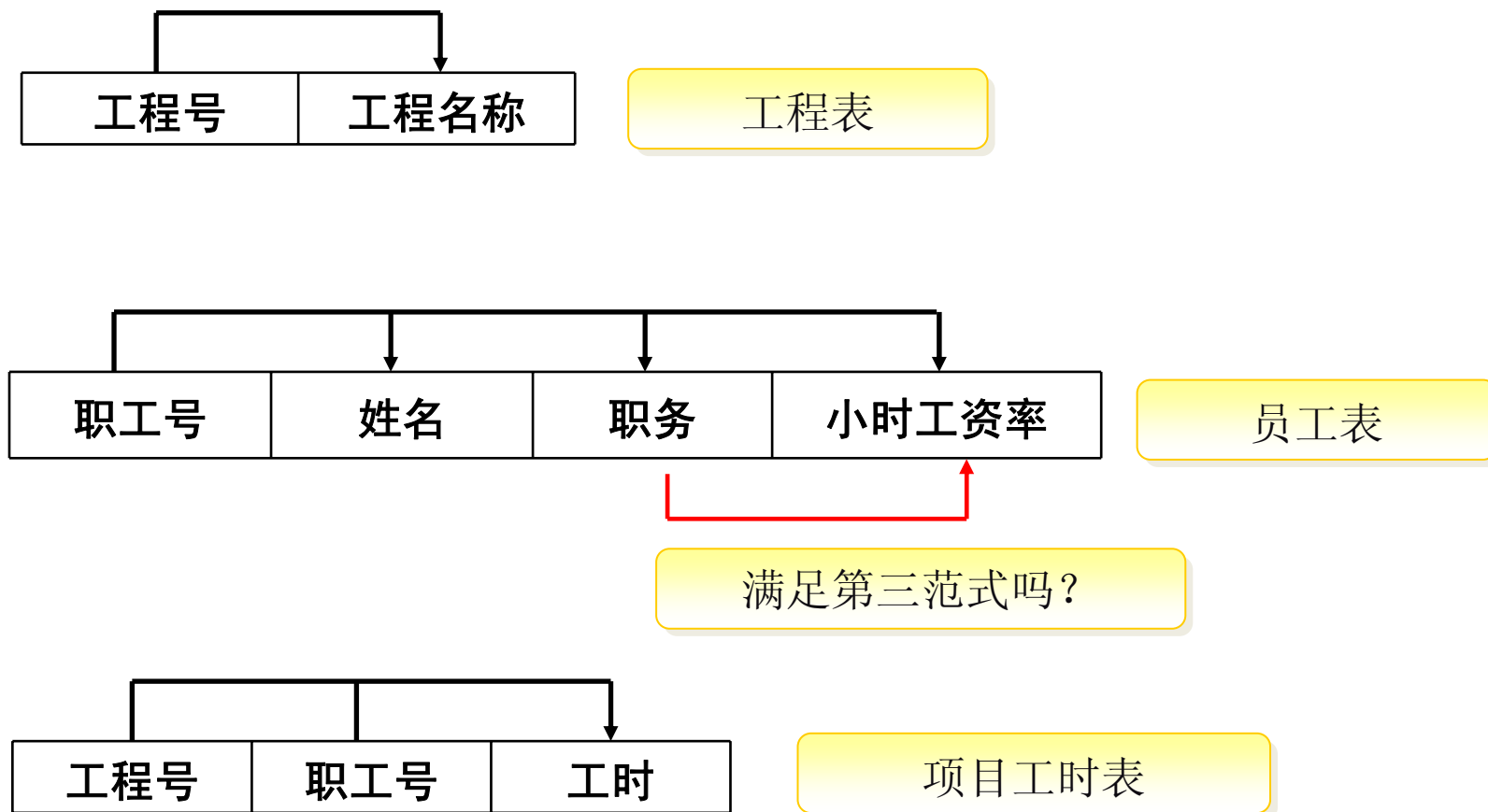
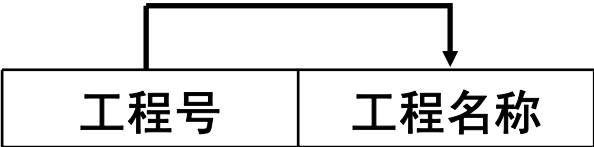
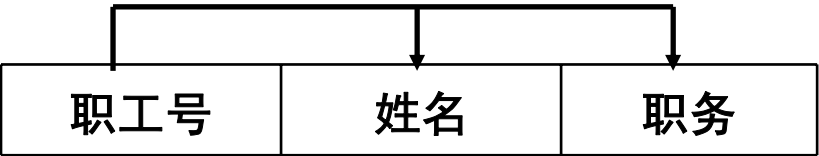


图-4 应用第二范式

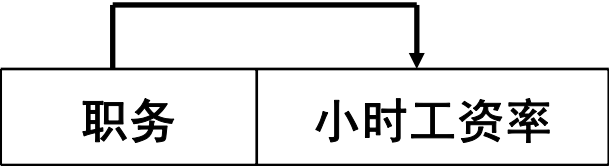
应用第三范式规范化



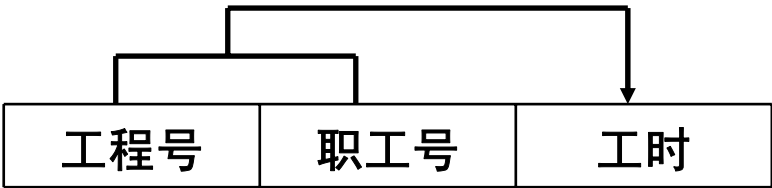
工程表



员工表



职务表



工程表

规范化和性能的关系

- 为满足某种商业目标，数据库性能比规范化数据库更重要
 - 通过在给定的表中添加额外的字段，以大量减少需要从中搜索信息所需的时间
 - 通过在给定的表中插入计算列（如成绩总分），以方便查询
- 进行规范化的同时，还需要综合考虑数据库的性能。

总结 1-1

- 在需求分析阶段，设计数据库的一般步骤为：
 - 收集信息
 - 标识对象
 - 标识每个对象的属性
 - 标识对象之间的关系
- 在概要设计阶段和详细设计阶段，设计数据库的步骤为：
 - 绘制E-R图
 - 将E-R图转换为表格
 - 应用三大范式规范数据库

总结 1-2

- 为了设计结构良好的数据库，需要遵守一些专门的规则，称为数据库的设计范式。
 - 第一范式（1NF）的目标：确保每列的原子性。
 - 第二范式（2NF）的目标：确保表中的每列，都和主键相关。
 - 第三范式（3NF）的目标：确保每列都和主键列直接相关，而不是间接相关。