第3章

数据库的设计

当今信息化时代几乎所有的信息系统都需要有数据库的支撑，如何针对具体应用环境，利用数据库管理系统构造最适合的数据库模式，建立数据库及其应用系统，对数据进行科学高效的管理，就是数据库设计的主要内容。数据库设计是将数据库系统与现实世界进行密切、有机和协调一致地结合的过程。因此，数据库设计者需要同时具备数据库管理系统及其实际应用对象两个方面的知识。

近年来，随着互联网普及率的提高，我国的电子商务产业得到快速发展。随着越来越多的电子商务平台的出现，网络购物也逐渐成为大众主流的购物方式。网络购物平台会涉及大量的客户与商品数据的处理，这些平台是如何对这些数据进行高效管理的呢？它又是如何对广大客户的消费情况进行分类统计的呢？本章我们将以网络购物系统为例，帮助读者了解数据库的设计方法，并在接下来的章节中逐步创建一个简易的网络购物系统，最终解开网络购物系统的神秘面纱。

3.1 关系数据库设计概述

关系数据库设计概述

人们通常把使用数据库的各类信息系统都称为数据库应用系统，如网络购物系统、工资管理系统、票务系统、银行系统、人脸识别系统等。

数据库设计从广义上来说，是数据库及其应用系统的设计，即设计整个数据库应用系统。从狭义上来说，是设计数据库本身，即设计数据库的各级模式并建立数据库，是数据库应用系统设计的一部分。本书讲解侧重广义的数据库设计，以网络购物系统为例讲解数据库及应用系统的设计。

数据库设计是构建数据库系统的重要环节。一个良好的数据库设计，可以为用户和系统管理人员提供清晰的数据逻辑关系和高效的数据管理效率。低劣的数据库设计，则会在数据的存储和管理过程中造成数据访问率低，存储空间浪费以及数据的插入、删除异常等问题。

良好的数据库设计表现在以下几个方面。

（1）访问效率高，使用方便，维护简单。

（2）数据冗余度低，存储空间小，便于进一步扩展。

（3）应用程序易于开发。

（4）基于有效安全机制的数据安全可以得到保障。

（5）数据的备份和恢复容易实现。

低劣的数据库设计往往表现在以下几个方面。

（1）数据的访问效率低下。

（2）大量的数据冗余造成存储空间浪费。

（3）数据的插入和删除出现异常等问题。

3.1.1　数据库设计方法和步骤

如果我们现在就开始设计一个网络购物数据库系统，应该从哪儿入手？按照什么样的方法进行设计？早期的数据库设计工作，都是依赖于设计人员的自身经验，缺乏成熟的科学理论和工程方法的支持，因此设计质量难以保证，数据库常常运行一段时间后又会发现不同程度的问题，需要不断修改甚至重新设计，大大增加了系统维护成本。

为此，人们多年来不断地探索，提出了各种数据库设计方法。其中，新奥尔良（New Orleans）方法是一种比较著名的数据库设计方法，它通过分步设计，遵循自顶向下、逐步求精的原则，将数据库设计过程分解为若干相互独立又相互依存的阶段，每一阶段采用不同的辅助工具，解决不同的问题，从而将问题局部化，减少局部问题对整体设计的影响。新奥尔良方法将数据库设计分为需求分析、概念结构设计、逻辑结构设计和物理结构设计 4 个阶段，如图 3.1 所示。

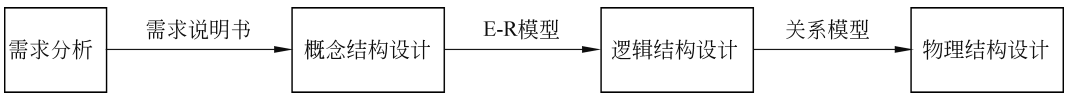


图 3.1　新奥尔良方法数据库设计步骤

其后，经过研究人员的不断改进，目前公认的比较权威的方法是将数据库系统设计分为 6 个阶段进行，分别是需求分析、概念结构设计、逻辑结构设计、物理结构设计、数据库实施和数据库运行及维护。

另外，我们在相关的文献或者资料中也会看到其他一些数据库设计方法，例如，基于 E-R 模型的设计方法、基于 3NF（第三范式）的设计方法、基于抽象语法规范的设计方法等，这些都是针对不同数据库设计阶段使用的具体技术和方法。

3.1.2　数据库设计过程

考虑到数据库开发的全过程，按照结构化设计的方法，将数据库设计分为 6 个阶段。

1. 需求分析

收集和分析用户需求，包括数据与处理。

2. 概念结构设计

对用户需求进行归纳和抽象，形成概念模型（例如 E-R 模型）。

3. 逻辑结构设计

将概念模型转换为某个数据库管理系统所支持的数据模型(例如关系模型),并对其进行优化。

4. 物理结构设计

为数据模型选取一个最适合应用环境的物理结构(包括存储结构和存取方法)。

5. 数据库实施

根据逻辑设计和物理设计的结果建立数据库,组织数据入库并进行试运行。

6. 数据库运行及维护

在数据库系统运行过程中不断对其进行调整、评价与修改。

设计一个完善的数据库系统不可能一蹴而就,它往往是上述 6 个阶段不断反复、逐步求精的过程。

3.2 关系数据库的设计

关系数据库的设计主要包括需求分析、概念结构设计、逻辑结构设计、物理结构设计、数据库实施、数据库运行及维护 6 个阶段,每个阶段都有不同的内容和任务,具体如下。

3.2.1 需求分析

需求分析是设计数据库的起点。需求分析是设计人员通过与用户交流和调查等,分析并逐步明确用户对系统的需求,包括数据需求、业务处理需求、安全性及完整性要求等。需求分析的结果是否准确将直接影响后面各个阶段的设计,并影响设计结果是否合理和实用。因此,需求分析是整个设计工作最基础、是最耗时的部分。

需求分析的重点是“数据”和“处理”,通过调查、收集与分析,获得用户对数据库的如下要求。

(1) 数据要求。即用户对需要处理的数据内容及性质的要求。

(2) 处理要求。指用户要求数据库需要具备的数据处理功能,以及用户对处理性能的要求。

(3) 安全性与完整性要求。

调查用户需求的具体步骤如下。

(1) 调查组织机构情况和各部门的业务活动情况。

(2) 在熟悉业务活动的基础上,协助用户明确对新系统的各种要求,包括数据要求、处理要求、安全性与完整性要求。

(3) 确定新系统的边界。基于前面的调查和分析,明确哪些功能由计算机完成或将

来准备让计算机完成，哪些活动由人工完成。由计算机完成的功能就是新系统应该实现的功能。

常用调查方法如下。

（1）跟班作业和业务咨询。通过亲身参加业务工作、与客户进行座谈、请专人介绍等来了解业务活动的情况。

（2）组织用户填写问卷调查。

（3）查阅记录。查阅与原系统有关的数据记录。

【例 3.1】　对网络购物系统进行需求分析。

（1）数据要求。

系统存储客户和商品的基本信息，详细记录客户的订单信息。

（2）处理要求。

① 客户可以查看自己的信息；可以查看不同商品的信息；能够下单购买商品；可以查询订单情况。

② 系统能够对商品的情况进行统计；能够对订单情况进行统计。

③ 系统可以加入新的客户信息；加入新的商品信息。

④ 系统可以删除过期商品信息。

⑤ 系统可以更新订单情况。

⑥ 系统可以查询客户及商品信息；可以查询订单情况；可以对商品和客户的购买情况进行分类统计。

（3）数据库安全性与完整性需求。

为保证数据库的安全，需要给不同用户设置不同的权限。客户对自己所购买商品有选择和查询的权限，可以查看自己的订单，但是对其他客户的订单无查看权限，也没有修改权限。

为了防止不符合规范的数据进入数据库，确保数据库中存储的数据正确、有效、相容，关系数据库必须满足关系的完整性约束条件。精确的需求分析，可以帮助我们在接下来的概念结构和逻辑结构设计阶段，准确设定关系表的主键、外键以及各个属性值的类型和取值范围，从而在实体完整性、参照完整性和用户定义完整性 3 方面满足数据库的完整性需求。

3.2.2　概念结构设计

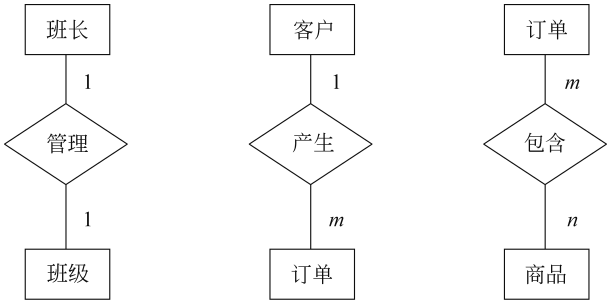
明确用户的需求之后，就要对用户需要处理的数据以及操作进行归纳和抽象，形成概念模型，这个过程就是概念结构设计。概念结构设计是设计人员从用户的视角，对信息进行抽象和描述。因此，概念模型应该具备真实、客观反映现实世界和易于理解的特点。目前最常使用的概念模型是 E-R 模型。在 2.1.1 节，我们介绍了概念模型的相关概念：实体、实体型、实体集、属性、联系，大家对概念模型有了初步的了解，下面来深入了解实体之间的联系。

实体之间的联系通常指不同实体集之间的联系。实体集之间的联系类型有如下 3 种。

(1) 一对一(1∶1)：如果实体集 A 中的每个实体，在实体集 B 中至多有一个(也可以没有)实体与之关联，反之亦然，则实体集 A 与实体集 B 为一对一联系，记为 1∶1。例如，班长与班级之间的联系：一个班长管理一个班级，每一个班级有一个班长，因此班长与班级之间的联系类型为 1∶1，实体集之间的联系可以用图表示，如图 3.2(a)所示。

(2) 一对多(1∶m)：实体集 A 中的每个实体在实体集 B 中有 m 个($m\geqslant 1$)实体与之关联，而实体集 B 中的每个实体在实体集 A 中最多只有一个实体与之关联，则实体集 A 与实体集 B 为一对多联系，记为 1∶m。例如，客户与订单之间的联系：一个客户可以下单多次，而每个订单只涉及一个客户，所以客户与订单之间的联系类型为 1∶m，如图 3.2(b)所示。

(3) 多对多(m∶n)：实体集 A 中的每个实体在实体集 B 中有 n 个($n\geqslant 1$)实体与之关联，实体集 B 中的每个实体在实体集 A 中有 m 个($m\geqslant 1$)实体与之关联，则实体集 A 与实体集 B 为多对多联系，记为 m∶n。例如，订单与商品之间的联系：一个订单会涉及多个商品，而每种商品会被多个订单包含，因此订单与商品之间的联系类型就是 m∶n，如图 3.2(c)所示。



(a) 1:1 联系　(b) 1:m 联系　(c) m:n 联系

图 3.2　两个实体间的 3 种联系类型

E-R 图是 E-R 模型图形化表示方法，它提供了实体集、属性和实体集之间联系的表示方法。

(1) 用矩形表示实体集，矩形框内标注实体集名称。

(2) 用椭圆形表示属性，椭圆形框内标注属性的名称，并用无向边将其与相应的实体连接起来。

(3) 用菱形表示实体集之间的联系，菱形框内标注联系的名称，并用无向边将其与所联系的实体集连接起来。

【例 3.2】　用 E-R 图来表示网络购物系统的概念模型。

(1) 实体集。

网络购物系统所涉及的实体集有客户、商品和订单。

(2) 属性。

① 客户实体的属性有客户编号、姓名、性别、注册日期、联系电话等。

② 商品实体的属性有商品编号、商品名称、类别、库存、是否上架、成本价格、销售价

格等。

③ 订单实体的属性有订单编号、客户编号、商品编号、订单时间、发货时间、配送地址和城市等。

(3) 联系。

① 每个客户会下多个订单，而每个订单只能对应一个客户，因此客户和订单之间是 1∶m 的联系。

② 每个订单可以涉及多种商品，而每种商品也可以包含在多个订单中，因此订单与商品之间是 m∶n 的联系。

通过初步分析，可以创建如图 3.3 所示的网络购物系统 E-R 图，受篇幅影响，这里只画出实体集部分属性。

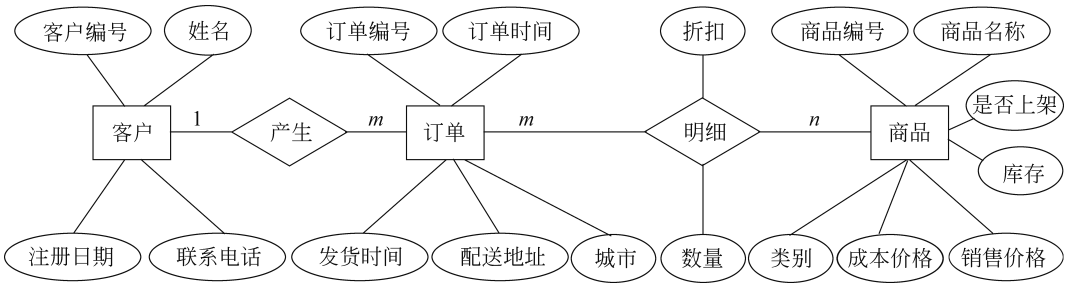


图 3.3　网络购物系统 E-R 图

3.2.3　逻辑结构设计

逻辑结构设计

逻辑结构设计是将概念结构设计阶段完成的 E-R 模型，转换成被选定的数据库管理系统支持的数据模型。由于目前的数据库应用系统大多采用支持关系模型的关系数据库管理系统，因此，这里主要讲 E-R 模型转换为关系模型的方法。关系模型通过关系(即二维表)表示实体集以及实体集之间的联系。E-R 模型包含实体集、属性、实体集之间的联系 3 部分内容。因此，将 E-R 模型转换成关系模型，实际上就是将实体集以及实体集之间的联系转换为一个个关系。转换规则如下。

(1) 实体集的转换：一个实体集转换成一个关系，实体的属性即为关系的属性，实体的码即为关系的码。

(2) 联系的转换：根据不同的联系类型进行不同的处理。

① 一对一的联系，可以转换为一个独立的关系，也可以与任意一端的关系合并。如果转换为一个独立的关系，则与该联系相连的各个实体的码以及该联系本身的属性作为关系的属性；如果与任意一端的关系合并，则需要加入另外一个实体的码和联系本身的属性作为此关系的属性。

② 一对多的联系，可以转换为一个独立的关系，也可以与多端的关系合并。如果转化为一个独立的关系，则与该联系相连的各实体的码以及联系本身的属性作为此关系的属性。若与多端的关系合并，需加入一端实体的码和联系的属性作为此关系的属性。

③ 多对多的联系，转换为一个独立的关系，其属性为多端实体的码加上联系本身的

属性。

3 个或 3 个以上的实体集间的一个多元联系,不管是何种联系类型,总是将多元联系类型转换成一个关系,其属性为与该联系相连的各实体的码及联系本身的属性。

【例 3.3】 将图 3.3 网络购物系统 E-R 图转化为关系模型。

客户实体集转换为一个单独的关系,名称为 customers。客户实体的属性有客户编号、姓名、性别、注册日期、联系电话,所以转换后的关系模式为 customers(customer_id,name,gender,registration_date,phone),主键为客户编号 customer_id。

商品实体集转换为一个单独的关系,名称为 items。商品实体的属性有商品编号、商品名称、类别、成本价格、销售价格、库存、是否上架。转换后的关系模式为 items(item_id,item_name,category,cost,price,inventory,is_online),主键为商品编号 item_id。

订单实体集转换为一个单独的关系,名称为 orders。订单实体的属性有订单编号、订单时间、配送地址、城市、发货时间。转换后的关系模式为 orders(order_id,order_date,address,city,shipping_date),主键为订单编号 order_id。

客户与订单之间为一对多的联系,该联系没有自己的属性,可以将其与多端关系合并,并将一端客户实体的码加入到关系中,即将其与 orders 关系合并,将客户实体的码 customer_id,加入到 orders 关系属性中。合并后 orders 关系模式变为 orders(order_id,customer_id,order_date,address,city,shipping_date),主键为 order_id。

订单与商品之间为多对多的联系,该联系的属性有数量和折扣,需要转换为一个单独的关系。其属性为多端实体的码加上本身的属性,即订单编号、商品编号、数量、折扣。转换后的关系模式为 order_details(order_id,item_id,discount,quantity),主键为(order_id,item_id)。

转换后,网络购物系统共包含 customers、orders、order_details、items 4 个关系表。

3.2.4 物理结构设计

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。针对已经确定的数据库的逻辑结构,选取一个最适合应用需求的物理结构的过程,就是数据库的物理结构设计,它依赖于选定的数据库管理系统。系统会根据数据的具体情况确定存储结构和存取方法。

物理结构设计过程中要对时间效率、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,数据库设计者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作为数据库的物理结构。如果评价结构满足原设计要求,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。

3.2.5 数据库实施

数据库实施阶段包括数据的载入与应用程序的编码和调试。

完成了数据库的逻辑结构设计和物理结构设计后,开发人员使用具体的 DBS 提供的数据定义语言(DDL)来描述和定义数据库结构。完成数据库定义后,还须装入各种实际

数据，具体的步骤包括：筛选数据、转换数据格式、输入数据和校验数据等。

在组织数据入库的同时还要调试应用程序，可使用模拟数据进行程序的调试。

3.2.6　数据库运行及维护

试运行阶段要重视数据库的校验工作。试运行过程中要测试各个功能是否满足设计的要求，对数据库的性能指标进行测试，分析其是否达到设计目标，未达到目标时需要针对数据库设计的各个阶段进行修改和调整，以满足用户的需求。

数据库系统经过试运行后，即可投入正式运行，在数据库系统运行过程中必须不断地对其进行评价、调整、修改。为了保证良好的应用效果，需要在运行过程中不断地对数据库进行优化和维护。

3.3　关系模型规范化设计

数据库设计的逻辑结构设计过程中，针对一个具体的应用，怎样才能构造一个合适的数据库模式，即应该构造几个关系模式，每个关系模式应该包含哪些属性等，这需要一定的衡量标准。关系模型的规范化理论就是一个数据库设计指导理论和衡量标准，它可以帮助我们设计出良好的关系模式并避免后续数据库操作过程中可能出现的问题。这里涉及范式的概念，不同的范式表示关系模式需要遵守的不同规则，在学习范式之前需要先了解函数依赖和关系模式中的键。

3.3.1　函数依赖

设关系模式 a_schema(customer_id, name, gender, address, item_id, order_date, shipping_date)，(customer_id, item_id)为主键，表 3.1 是关系模式某一时刻的数据表。

表 3.1　a_schema 关系模式的部分数据示例

customer_id	name	gender	address	item_id	order_date	shipping_date
107	林琳	女	武清区流星花园 6-6-66	b001	2018-6-18 18:00:02	2018-6-19 16:40:26
107	林琳	女	武清区流星花园 6-6-66	b002	2018-6-18 18:00:02	2018-6-19 16:40:26
107	林琳	女	武清区流星花园 6-6-66	sm01	2018-6-18 18:00:02	2018-6-19 16:40:26
106	孙丽娜	女	道里区和谐家园 66-66-666	f001	2018-11-11 17:10:21	2018-11-13 16:20:20
106	孙丽娜	女	道里区和谐家园 66-66-666	sm01	2018-11-11 17:10:21	2018-11-13 16:20:20
104	赵文博	男	海淀区清河小营东路 12 号学 9 公寓	s002	2019-6-18 19:01:32	2019-6-19 08:50:20

观察表 3.1 的数据可以发现这个关系模式存在如下问题。

(1) 数据冗余问题：客户的基本信息(包括客户姓名、性别)以及配送地址和订单时间有冗余。一个客户一次订单购买多少种不同的商品，这个客户对应的姓名、性别、配送地址、订单时间、发货时间信息就会重复多少次。

(2) 数据更新问题：如果一个客户的某个基本信息发生了变化，那么该客户购买了几种商品我们就需要更改多少次该客户基本信息，从而使修改变得烦琐，也很容易在修改过程遗漏部分信息，这样还会造成信息不一致的后果。

(3) 数据插入问题：如果购物平台有了新的客户加入，即已经有了客户基本信息，但是我们也不能把该客户加入到这个表中，因为客户没有购物，他的 item_id 是空的，而作为主键的 item_id，是不允许为空的。

(4) 数据删除问题：如果一个客户只购买过一种商品，后来又退货了，在删掉该客户的购买记录的同时，这个客户的基本信息也被删除掉了。

基于以上种种问题，我们可以看出，该关系模式不是一个好的关系模式，究其原因则是因为这个关系模式中某些属性存在“不良”的函数依赖关系，下面来介绍函数依赖的概念。

函数依赖的定义：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，如果 r 中两个元组在 X 上的属性值相等，在 Y 上的属性值也一定相等，则称 X 函数确定 Y 或 Y 函数依赖于 X，记作 $X \rightarrow Y$。其中，X 叫作决定因素，Y 叫作依赖因素。

跟函数依赖相关的一些术语和记号如下。

(1) $X \rightarrow Y$，但 $Y \not\subseteq X$(Y 不包含于 X)，则称 $X \rightarrow Y$ 是非平凡的函数依赖。

(2) $X \rightarrow Y$，但 $Y \subseteq X$(Y 包含于 X)，则称 $X \rightarrow Y$ 是平凡的函数依赖。因为平凡的函数依赖总是成立的，所以若不特别声明，本书后面提到的函数依赖，都指非平凡的函数依赖。

(3) 若 $X \rightarrow Y$，$Y \rightarrow X$，则记作 $X \leftrightarrow Y$。

(4) 若 Y 不函数依赖于 X，则记作 $X \not\rightarrow Y$。

(5) 如果 $X \rightarrow Y$，且对于 X 的任何一个真子集 X'，都有 $X' \not\rightarrow Y$，则称 Y 对 X 完全函数依赖，记作 $X \xrightarrow{F} Y$。

(6) 若 $X \rightarrow Y$，如果存在 X 的某一真子集 X'，使 $X' \rightarrow Y$，则称 Y 对 X 部分函数依赖，记作 $X \xrightarrow{P} Y$。

(7) 如果 $X \rightarrow Y$(非平凡函数依赖，且 $Y \not\rightarrow X$)，$Y \rightarrow Z$，则称 Z 传递函数依赖于 X，记作 $X \xrightarrow{\text{传递}} Z$。

【例 3.4】 关系模式 a_schema(customer_id, item_id, name, quantity)，主键为(customer_id, item_id)，存在以下函数依赖关系：

customer_id → name(姓名函数依赖于客户编号)

所以存在以下部分函数依赖：

(customer_id, item_id) $\xrightarrow{P}$ name(姓名部分函数依赖于客户编号和商品编号)

【例 3.5】　关系模式 order_details(order_id,item_id,discount,quantity),主键为(order_id,item_id),存在以下函数依赖关系:

$$(\text{order\_id},\text{item\_id}) \xrightarrow{F} \text{discount}(\text{折扣完全函数依赖于订单编号和商品编号})$$

【例 3.6】　设关系模式 b_schema(customer_id,city,province),主键为 customer_id,存在以下函数依赖关系:

$$\text{customer\_id} \rightarrow \text{city}(\text{城市函数依赖于客户编号})$$
$$\text{city} \rightarrow \text{province}(\text{省份函数依赖于城市})$$

所以有:

$$\text{customer\_id} \xrightarrow{\text{传递}} \text{province}(\text{省份通过城市传递函数依赖于客户编号})$$

3.3.2　关系模式中的键

键(码)是关系模式中一个很重要的概念,第 2 章已经给出了键的若干定义,这里用函数依赖的概念来定义键。

设 U 表示关系模式 R 的属性全集,即 $U=R\{A_1,A_2,\cdots,A_n\}$,$F$ 表示关系模式 R 上的函数依赖集,则关系模式可以表示为 $R(U,F)$。

1. 候选键

设 K 为关系模式 $R(U,F)$ 的属性或属性组,若 $K\rightarrow U$,则 K 为 R 的候选键。

2. 主键

如果关系模式 $R(U,F)$ 中有多个候选键,选其中的一个作为主键。

3. 全键

如果候选键为整个属性组则称为全键。

4. 外键

一个关系模式 $R(U,F)$ 中的某个属性(组)不是 R 的主键,但它是另外一个关系的主键,则该属性(组)称为关系 R 的外键。

5. 主属性

关系模式 $R(U,F)$ 中,包含在任一候选键中的属性称为主属性。

6. 非主属性

关系模式 $R(U,F)$ 中,不包含在任一候选键中的属性称为非主属性。

例如,关系模式 order_details(order_id,item_id,discount,quantity)中,order_id 不是主键,但它是关系模式 orders(order_id,customer_id,order_date,address,city,shipping_date)的主键。因此,order_id 是 order_details 的外键,同理,item_id 也是 order_details 的

外键。

【例 3.7】 关系模式 customers(customer_id,name,gender,phone),设不同的客户联系电话不同,则该关系模式中,有

候选键:customer_id,phone。

主键:customer_id 或者 phone。

主属性:customer_id,phone。

非主属性:name,gender。

【例 3.8】 关系模式 order_details(order_id,item_id,discount,quantity),则该关系模式中,有

候选键:(order_id,item_id)。

主键:(order_id,item_id)。

主属性:order_id,item_id。

非主属性:discount,quantity。

外键:order_id,item_id。

【例 3.9】 设有关系模式 c_schema(customer_id,item_id,order_date),如果规定每一个客户同一种商品可以多次购买,则该关系模式中,有

候选键:(customer_id,item_id,order_date)。

主键:也是该候选键。

主属性:customer_id,item_id,order_date。

非主属性:无。

外键:customer_id,item_id。

这种候选键为全部属性的表称为全键表。

3.3.3 范式

范式

关系数据库中的关系模式需要满足一定的要求,不同的要求对应不同的范式。关系模式按其规范化程度从低到高可分为 5 级范式。满足最低要求的关系模式称为第一范式,简称 1NF。在第一范式中又满足一些要求的称为第二范式,简称 2NF,以此类推,还有 3NF、BCNF、4NF、5NF。

所有范式中,只要关系模式满足第一范式,它就是合法的、允许的。后来,人们发现某些关系模式存在插入、删除异常,以及冗余度高、修改复杂等问题,因此开始研究关系模式的规范化问题。Codd 在 1971 年到 1972 年提出 1NF、2NF、3NF 的概念,1974 年 Codd 和 Boyce 共同提出了新的范式 BCNF,后来又有研究人员相继提出了 4NF、5NF。规范化程度较高者必是较低者的子集。各种范式之间的关系如下:

$$5NF \subset 4NF \subset BCNF \subset 3NF \subset 2NF \subset 1NF$$

数据库设计中,关系模式应该规范到第几范式需要根据实际情况确定,以高效、便捷和数据库操作出现错误概率小为标准,一般情况下规范到第三范式即可,BCNF、4NF、5NF 本书不做详细介绍。

1. 第一范式

定义：每个属性均不能再分解的关系模式。它是关系模式最基本的规范形式。如果关系模式 R 为第一范式则记作 $R \in 1NF$。

第一范式要求每个属性必须是不可分的数据项。图 3.4 中“订单情况”不是基本数据项，它是由两个基本数据项“订单时间”和“发货时间”组成的一个复合数据项。因此，这个关系模式不符合第一范式。非第一范式的关系转换成第一范式关系只需要将所有数据项都表示为不可分的最小数据项即可，如图 3.5 所示。

客户编号	订单情况	
	订单时间	发货时间
105	2018-5-6 12:10:20	2018-5-7 10:29:35
106	2018-11-11 17:10:21	2018-11-13 16:20:20
107	2018-6-18 18:00:02	2018-6-19 16:40:26

图 3.4　不符合第一范式的关系

客户编号	订单时间	发货时间
105	2018-5-6 12:10:20	2018-5-7 10:29:35
106	2018-11-11 17:10:21	2018-11-13 16:20:20
107	2018-6-18 18:00:02	2018-6-19 16:40:26

图 3.5　符合第一范式的关系

2. 第二范式

定义：如果关系模式 $R(U,F) \in 1NF$，且 R 中的每个非主属性完全函数依赖于 R 的候选键，则 R 为第二范式，记作 $R \in 2NF$。

例如，关系模式 d_schema(order_id,item_id,order_date,quantity)，候选键为(order_id,item_id)。该关系模式存在以下函数依赖关系：

order_id → order_date(订单时间函数依赖于订单编号)

所以存在部分函数依赖关系：

(order_id,item_id) $\xrightarrow{P}$ order_date(订单时间部分函数依赖于候选键)

因此，该关系模式不符合 2NF。关系模式不符合 2NF 就会产生插入和删除异常以及修改复杂的问题。

可以通过模式分解将低一级范式的关系模式转换为高一级范式的关系模式集合。我们可以将关系模式 d_schema 分解为 d_schema(order_id,item_id,quantity)和 e_schema(order_id,order_date)两个关系模式。分解后的这两个关系模式都符合 2NF。

3. 第三范式

定义：如果关系模式 $R(U,F) \in 2NF$，且每个非主属性都不传递函数依赖于候选键，

则 R 满足第三范式,记作 $R \in 3NF$。

例如,关系模式 f_schema(order_id,city,province),候选键为 order_id,存在如下函数依赖关系:

$$\text{order\_id} \rightarrow \text{city}(\text{城市函数依赖于订单编号})$$

$$\text{city} \rightarrow \text{province}(\text{省份函数依赖于城市})$$

所以有以下传递函数依赖关系:

$$\text{order\_id} \xrightarrow{\text{传递}} \text{province}(\text{省份通过城市传递函数依赖于订单编号})$$

因此,该关系模式不满足 3NF,从而也会导致数据插入、删除操作异常,以及修改复杂的问题。可以将此关系模式分解为 f_schema(order_id,city)和 g_schema(city,province),分解后的两个关系均不存在非主属性对候选键的传递函数依赖,符合第三范式。

3.3.4 关系模式的规范化

对于规范化程度低的关系模式,可以将其分解为若干个规范化程度高的关系模式,以此提高关系模式的规范化程度。分解后的关系模式从语义上来说,每个关系只能描述一个主题,如果描述了多个主题,这个关系还要进行进一步分解。分解后的模式应该与原来的模式等价,不能在规范化过程中消除一个问题的同时又产生其他问题,因此模式分解必须遵守以下两个原则。

(1) 模式分解具有无损连接性。

(2) 模式分解保持函数依赖。

无损连接是指分解后的关系模式通过自然连接能够恢复到原来的关系,恢复后既不会增加信息也不会减少信息。

保持函数依赖是指分解过程中原来关系模式中的函数依赖不能丢失,也就是分解前后关系模式的语义应该保持一致。

【例 3.10】 关系模式 h_schema(order_id,item_id,customer_id,name,discount,address,city,order_date),某个时刻数据表如表 3.2 所示,对其进行规范化处理。

表 3.2 关系模式 h_schema 某时刻数据表

order_id	item_id	customer_id	name	discount	address	city	order_date
1	sm01	105	Adrian_Smith	0.85	海淀区西三旗幸福小区 60 号楼 6 单元 606	北京市	2018-5-6 12:10:20
3	b001	107	林琳	0.80	武清区流星花园 6-6-66	天津市	2018-6-18 18:00:02
3	b002	107	林琳	0.85	武清区流星花园 6-6-66	天津市	2018-6-18 18:00:02
3	sm01	107	林琳	0.90	武清区流星花园 6-6-66	天津市	2018-6-18 18:00:02

续表

order_id	item_id	customer_id	name	discount	address	city	order_date
4	f001	106	孙丽娜	0.80	道里区和谐家园 66-66-666	哈尔滨市	2018-11-11 17:10:21
4	sm01	106	孙丽娜	0.90	道里区和谐家园 66-66-666	哈尔滨市	2018-11-11 17:10:21
4	sm02	106	孙丽娜	0.90	道里区和谐家园 66-66-666	哈尔滨市	2018-11-11 17:10:21
5	m001	103	Grace_Brown	0.90	市北区幸福北里 88 号院	青岛市	2019-2-1 17:51:01
5	sm01	103	Grace_Brown	0.90	市北区幸福北里 88 号院	青岛市	2019-2-1 17:51:01
6	s002	104	赵文博	0.90	海淀区清河小营东路 12 号学 9 公寓	北京市	2019-6-18 19:01:32

从关系模式的各个属性来看，它们都是最小的数据项，不可以再分解，因此该关系模式符合 1NF。关系模式的候选键有 order_id 和(customer_id，item_id，order_date)，由于客户编号不同，姓名也不会相同，因此存在以下函数依赖关系：

$$\text{customer\_id} \rightarrow \text{name}$$

所以存在以下部分函数依赖关系：

$$(\text{customer\_id},\text{item\_id},\text{order\_date}) \xrightarrow{P} \text{name}$$

因此，该关系模式不符合 2NF。从表中的数据就可以看出有大量冗余信息。例如 name、address、city 等属性值中就存在大量冗余。可以通过模式分解对其进行规范化处理。

通过分析，可以看出该关系模式既包含了对客户的描述(customer_id，name)，也包含了对订单的描述(order_id，customer_id，address，city，item_id，quantity，order_date)，根据每个关系模式尽量描述一个实体或实体之间的联系的原则，对其进行模式分解。可以分解为以下两个关系模式：

customers(customer_id，name)

orders(order_id，customer_id，order_date，item_id，discount，address，city)

分解后的 customers 关系模式候选键为 customers_id，不存在部分函数依赖关系，符合 2NF。而在 orders 关系模式中，(customer_id，item_id，order_date)是其中的一个候选键，由于折扣取决于商品编号和订单编号，因此该关系模式存在部分函数依赖关系：

$$(\text{customer\_id},\text{item\_id},\text{order\_date}) \rightarrow \text{discount}$$

所以需要将 orders 关系模式继续分解。可以将其中涉及的每个订单的详细信息分离出来，形成 order_details 关系模式。因此，orders 关系模式分解为以下两个关系模式：

orders(order_id，customer_id，order_date，address，city)

order_details(order_id，item_id，discount)

分解之后的两个关系模式不再存在部分函数依赖关系。

如表 3.2 所示,分解之后的 3 个关系表某时刻数据如表 3.3～表 3.5 所示。将分解前后的数据进行对比可以看出,分解之后的关系模式大大降低了数据冗余度,数据表也变得简洁、清晰了。

表 3.3 customers 某个时刻数据表

customer_id	name	customer_id	name
105	Adrian_Smith	103	Grace_Brown
107	林琳	104	赵文博
106	孙丽娜		

表 3.4 orders 某个时刻数据表

order_id	customer_id	address	city	order_date
1	105	海淀区西三旗幸福小区 60 号楼 6 单元 606	北京市	2018-5-6 12:10:20
3	107	武清区流星花园 6-6-66	天津市	2018-6-18 18:00:02
4	106	道里区和谐家园 66-66-666	哈尔滨市	2018-11-11 17:10:21
5	103	市北区幸福北里 88 号院	青岛市	2019-2-1 17:51:01
6	104	海淀区清河小营东路 12 号学 9 公寓	北京市	2019-6-18 19:01:32

表 3.5 orders_details 某个时刻数据表

order_id	item_id	discount	order_id	item_id	discount
1	sm01	0.85	4	sm01	0.90
3	b001	0.80	4	sm02	0.90
3	b002	0.85	5	m001	0.90
3	sm01	0.90	5	sm01	0.90
4	f001	0.80	6	s002	0.90

分解之后的 customers、order_details 两个关系模式中不存在传递函数依赖,因此,都符合 3NF。而关系模式 order_details 中,由于城市取决于配送地址,即:

$$\text{address} \rightarrow \text{city}(\text{城市函数依赖于配送地址})$$

因此,orders 关系模式存在以下传递依赖关系:

$$\text{order\_id} \xrightarrow{\text{传递}} \text{city}(\text{城市通过配送地址传递函数依赖于订单编号})$$

所以,orders 不符合 3NF。

但是考虑到实际需求,购买商品后续处理过程中需要同时兼顾地址和城市,因此,关系模式不需要再接着进行分解,可以通过设置用户自定义完整性来解决传递函数依赖带来的问题。

如果将分解后的表 3.3 与表 3.4、表 3.5 进行自然连接，会发现连接后得到的结果与表 3.2 中的数据一致，没有信息的增加或者丢失。因此，自然连接后的关系模式又恢复成了原来的关系模式，此模式分解满足无损连接的要求。将未分解前的关系模式 orders 的主键定为 order_id，通过分析可以看出，分解前后的函数依赖关系一致，说明分解后保持了原有的函数依赖关系。因此，此模式分解既满足了无损连接性又保持了原有函数依赖关系，符合要求，是一个有效的分解。

一般情况下，在进行模式分解时，我们应将有直接依赖关系的属性放置在一个关系模式中，这样得到的结果既能保持无损连接性，又能保持原有函数依赖关系不变。

知识点小结

关系数据库设计内容主要包括需求分析、概念结构设计、逻辑结构设计、物理结构设计、数据库实施和数据库运行及维护。关系模型的规范化理论是数据库设计指导理论和衡量标准，有助于设计出良好的关系模式，并避免后续数据库操作过程中可能出现的问题。不同的范式表示关系模式需要遵守的不同规则，具体数据库设计需要满足第几范式应该根据实际情况确定，以高效、便捷和错误概率低作为标准。

习　　题

一、选择题

1. 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，如果 r 中两个元组在 X 上的属性值相等，在 Y 上的属性值也一定相等，则称(　　)。

A. Y 函数决定 X　　B. X 函数决定 Y

C. X 函数依赖于 Y　　D. 以上说法都不对

2. 生成 DBMS 支持的关系模型是在(　　)阶段完成的。

A. 数据库概念结构设计　　B. 数据库逻辑结构设计

C. 数据库物理设计　　D. 数据库运行及维护

3. 在关系模式 $R(U)$ 中，X、Y、Z 是 R 的 3 个不同的属性或属性组，如果 $X \rightarrow Y$(Y 不是 X 的子集)，且 $Y \rightarrow Z$，则称 Z 对 X(　　)。

A. 传递函数依赖　　B. 部分函数依赖

C. 完全函数依赖　　D. 直接函数依赖

4. 在数据库设计中，创建 E-R 图的过程属于(　　)阶段。

A. 概念结构设计　　B. 逻辑结构设计

C. 物理结构设计　　D. 程序结构设计

5. 如果在一个关系 R 中，每个数据项都是不可分割的，那么它一定符合(　　)。

A. 1NF　　B. 2NF　　C. 3NF　　D. 4NF

6. 任何一个满足 2NF 但不满足 3NF 的关系模式都存在(　　)。

A. 主属性对候选键的部分依赖　　B. 非主属性对候选键的部分依赖

C. 主属性对候选键的传递依赖　　D. 非主属性对候选键的传递依赖

7. 关系数据库规范化是为解决关系数据库中(　　)问题而引入的。

A. 插入、删除和数据冗余　　B. 提高查询速度

C. 减少数据操作的复杂性　　D. 保证数据的安全性和完整性

8. 关系模型中的关系模式至少符合(　　)。

A. 1NF　　B. 2NF　　C. 3NF　　D. BCNF

9. 在数据库设计中,将 E-R 图转换成关系模型的过程称为(　　)。

A. 概念结构设计　　B. 逻辑结构设计

C. 物理结构设计　　D. 程序结构设计

二、简答题

1. 有关系模式:职工项目(部门编号,部门名称,员工编号,员工姓名,项目编号,项目名称,加入项目的日期),请解答以下问题。

(1) 确定该关系模式的关键字、范式等级。

(2) 若不满足 3NF,则将其化为 3NF。

【注】 其中,每个职工属于不同的部门,每个职工可以加入不同的项目。

2. 某关系表如表 3.6 所示,该关系是否满足 1NF? 若不满足请将其化为符合 1NF 的关系。

表 3.6　职工项目关系模式

考生编号	姓名	性别	考生学校	考场号	考场地点	成绩	
						考试成绩	平时成绩

3. 图 3.6 为一个学生选课系统 E-R 图,其中涉及学生和课程两个实体,实体属性以及联系如图 3.6 所示。

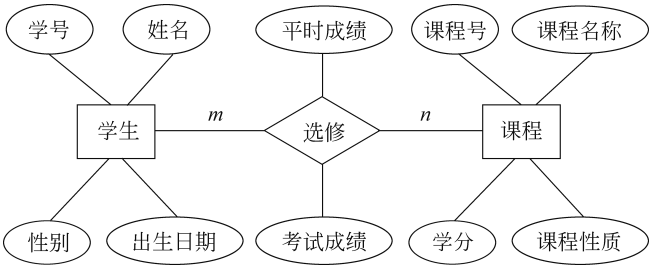


图 3.6　学生选课 E-R 图

请将该 E-R 图转换为关系模式。