第七章

数据库设计

是指对于一个给定的应用环境,构造优化的数据库逻辑模式和物理结构,并据此建立数据库及其应用系统,使之能够有效地存储和管理数据,满足各种用户的应用需求(信息要求和处理要求)
目标:为用户和各种应用系统提供一个信息基础设施和高效率的运行环境
在数据库领域内,常常把使用数据库的各类系统统称为数据库应用系统。

数据库设计的特点

数据库建设是硬件、软件和干件的结合

三分技术,七分管理,十二分基础数据
技术与管理的界面称之为“干件”

数据库设计应该与应用系统设计相结合

结构(数据)设计:设计数据库框架或数据库结构
行为(处理)设计:设计应用程序、事务处理等

结构和行为分离的设计

传统的软件工程忽视对应用中数据语义的分析和抽象,只要有可能就尽量推迟数据结构设计的决策
早期的数据库设计致力于数据模型和建模方法研究,忽视了对行为的设计

数据库设计方法

手工与经验相结合方法

设计质量与设计人员的经验和水平有直接关系
缺乏科学理论和工程方法的支持,工程的质量难以保证
数据库运行一段时间后常常又不同程度地发现各种问题,增加了维护代价

规范设计法

手工设计方法
基本思想
过程迭代和逐步求精
典型方法
新奥尔良方法
将数据库设计分为若干阶段和步骤
基于 E-R 模型的数据库设计方法
概念设计阶段广泛采用

3NF的设计方法

逻辑阶段可采用的有效方法

ODL方法

面向对象的数据库设计方法

数据库设计的基本步骤

准备工作

1.选定参加设计的人员

数据库分析设计人员

数据库设计的核心人员

自始至终参与数据库设计

其水平决定了数据库系统的质量

用户

在数据库设计中也是举足轻重的

主要参加需求分析和数据库的运行维护

用户积极参与带来的好处(加速数据库设计,提高数据库设计的质量)

程序员

在系统实施阶段参与进来,负责编制程序

操作员

在系统实施阶段参与进来,准备软硬件环境

设计过程(六个阶段)

需求分析阶段

准确了解与分析用户需求(包括数据与处理)

是整个设计过程的基础,是最困难、最耗费时间的一步

概念结构设计阶段

是整个数据库设计的关键

通过对用户需求进行综合、归纳与抽象,形成一个独立于具体 DBMS 的概念模型

逻辑结构设计阶段

将概念结构转换为某个 DBMS 所支持的数据模型

对其进行优化

数据库物理设计阶段

为逻辑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)

数据库实施阶段

运用 DBMS 提供的数据语言、工具及宿主语言,根据逻辑设计和物理设计的结果

建立数据库

编制与调试应用程序

组织数据入库

进行试运行

数据库运行和维护阶段

数据库应用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、调整与修改。

设计特点

在设计过程中把数据库的设计和对数据库中数据处理的设计紧密结合起来

将这两个方面的需求分析、抽象、设计、实现在各个阶段同时进行,相互参照,相互补充,以完善两方面的设计

设计过程各个阶段的设计描述:

设计阶段	数据	处理
需求分析	数据字典，全系统中数据项， 数据流，数据存储的描述	数据
概念结构设计	概念模型(E-R 图),数据字典	系统说明书包括:新系统要求、方案和概图; 反映新系统信息流的数据流图
逻辑结构设计	某种数据模型关系	系统结构图(模块结构)
物理设计	存储安排，方法选择， 存取路径建立	模块设计，IPO表
实施阶段	编写模式，装入数据， 数据库试运行	程序编码，编译链接，测试
运行维护	性能监测，转储/恢复， 数据库重组和重构	新旧系统转换，运行，维护(修正性， 适应性，改善性维护)

数据库各级模式的形成过程

需求分析阶段

综合各个用户的应用需求

概念设计阶段

形成独立于机器特点,独立于各个DBMS 产品的概念模式(E-R图)

逻辑设计阶段

首先将 E-R 图转换成具体的数据库产品支持的数据模型,如关系模型,形成数据库逻辑模式,然后根据用户处理的要求、安全性的考虑,在基本表的基础上再建立必要的视图(View) ,形成数据的外模式

物理设计阶段

根据 DBMS 特点和处理的需要,进行物理存储安排,建立索引,形成数据库内模式

需求分析

任务

通过详细调查现实世界要处理的对象(组织、部门、企业等),充分了解原
系统(手工系统或计算机系统)工作概况,明确用户的各种需求。在此基础上确定新系统的功能。新系统必须充分考虑今后可能的扩充和改变,不能仅仅按当前应用需求来设计数据库

重点

调查、收集与分析用户在数据管理中的信息要求、处理要求、安全性与完
整性要求。

信息要求

用户需要从数据库中获得信息的内容与性质

由用户的信息要求可以导出数据要求,即在数据库中需要存储哪些数据

处理要求

对处理功能的要求

对处理的响应时间的要求

对处理方式的要求 (批处理 / 联机处理)

新系统的功能必须能够满足用户的信息要求、处理要求、安全性与完整性要求。

难点

用户缺少计算机知识,开始时无法确定计算机究竟能为自己做什么,不能做什么,因此无法一下子准确地表达自己的需求,他们所提出的需求往往不断地变化。

设计人员缺少用户的专业知识,不易理解用户的真正需求,甚至误解用户的需求。

新的硬件、软件技术的出现也会使用户需求发生变化。

解决办法

设计人员必须采用有效的方法,与用户不断深入地进行交流,才能逐步得以确定用户的实际需求

方法

调查清楚用户的实际需求并进行初步分析

与用户达成共识

进一步分析与表达这些需求

数据字典

用途

是各类数据描述的集合
是进行详细的数据收集和数据分析所获得的主要结果
在数据库设计中占有很重要的地位

内容

数据项(数据的最小组成单位)
数据结构
数据流
数据存储
处理过程

数据项

不可再分的数据单位
数据项描述={数据项名,数据项含义说明,别名,数据类型,长度,取值范围,
取值含义,与其他数据项的逻辑关系,数据项之间的联系}
取值范围、与其他数据项的逻辑关系定义了数据的完整性约束条件

数据结构

反映了数据之间的组合关系
一个数据结构可以由若干个数据项组成,也可以由若干个数据结构组成,或由若干个数据项和数据结构混合组成。
数据结构描述={数据结构名,含义说明,组成 : {数据项或数据结构}}

数据流

是数据结构在系统内传输的路径
数据流描述={数据流名,说明,数据流来源,数据流去向,组成 : {数据结构},平均流量,高峰期流量}
来源是说明该数据流来自哪个过程
去向是说明该数据流将到哪个过程去
平均流量是指在单位时间(每天、每周、每月等)里的传输次数
高峰期流量则是指在高峰时期的数据流量

数据存储

是数据结构停留或保存的地方,也是数据流的来源和去向之一。
数据存储描述={数据存储名,说明,编号,流入的数据流,流出的数据流,组成 : {数据结构},数据量,存取频度,存取方式}
流入的数据流:指出数据来源
流出的数据流:指出数据去向
数据量:每次存取多少数据,每天(或每小时、每周等)存取几次等信息
存取方式:批处理 / 联机处理;检索 / 更新;顺序检索 / 随机检索

处理过程

处理过程的具体处理逻辑一般用判定表或判定树来描述。数据字典中只需要描述处理过程的说明性信息
处理过程描述={处理过程名,说明,输入 : {数据流},输出 : {数据流},处理 : {简要说明}}
简要说明:主要说明该处理过程的功能及处理要求
功能:该处理过程用来做什么
处理要求:处理频度要求(如单位时间里处理多少事务,多少数据量);响应时间要求等
处理要求是后面物理设计的输入及性能评价的标准

小结

数据字典是关于数据库中数据的描述,是元数据,而不是数据本身
数据字典在需求分析阶段建立,在数据库设计过程中不断修改、充实、完善
设计人员应充分考虑到可能的扩充和改变,使设计易于更改,系统易于扩充
必须强调用户的参与

概念结构

需求分析阶段描述的用户应用需求是现实世界的具体需求，将需求分析得到的用户需求抽象为信息结构即概念模型的过程就是概念结构设计.
概念结构是各种数据模型的共同基础,它比数据模型更独立于机器、更抽象,从而更加稳定。
概念结构设计是整个数据库设计的关键.

特点

１．能真实、充分地反映现实世界,包括事物和事物之间的联系,能满足用户对数据的处理要求。是对现实世界的一个真实模型。
２．易于理解,从而可以用它和不熟悉计算机的用户交换意见,用户的积极参与是数据库设计成功的关键。
３．易于更改,当应用环境和应用要求改变时,容易对概念模型修改和扩充。
４．易于向关系、网状、层次等各种数据模型转换。

设计概念结构的四类方法

1.自顶向下(首先定义全局概念结构的框架,然后逐步细化)
2.自底向上(首先定义各局部应用的概念结构,然后将它们集成起来,得到全局概念结构)
3.逐步扩张(首先定义最重要的核心概念结构,然后向外扩充,以滚雪球的方式逐步生成其他概念结构,直至总体概念结构)
4.混合策略(将自顶向下和自底向上相结合,用自顶向下策略设计一个全局概念结构的框架,以它为骨架集成由自底向上策略中设计的各局部概念结构。)

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

自底向上设计概念结构的步骤

1.抽象数据并设计局部视图
2.集成局部视图,得到全局概念结构

数据抽象

概念结构是对现实世界的一种抽象

1.从实际的人、物、事和概念中抽取所关心的共同特性,忽略非本质的细节
2.把这些特性用各种概念精确地加以描述
3.这些概念组成了某种模型

三种常用抽象

1.分类
定义某一类概念作为现实世界中一组对象的类型
这些对象具有某些共同的特性和行为
它抽象了对象值和型之间的“ is member of” 的语义
在 E-R 模型中,实体型就是这种抽象
2.聚集
定义某一类型的组成成分
它抽象了对象内部类型和成分之间“ is part of” 的语义
在 E-R 模型中若干属性的聚集组成了实体型,就是这种抽象
3.概括
定义类型之间的一种子集联系
它抽象了类型之间的“ is subset of” 的语义
概括有一个很重要的性质:继承性。子类继承超类上定义的所有抽象

用途

对需求分析阶段收集到的数据进行分类、组织(聚集),形成
1.实体
2.实体的属性,标识实体的码
3.确定实体之间的联系类型(1:1 , 1:n , m:n)

局部视图设计

选择局部应用

需求分析阶段,已用多层数据流图和数据字典描述了整个系统。
设计分 E-R 图首先需要根据系统的具体情况,在多层的数据流图中选择一个适当层次的数据流图,让这组图中每一部分对应一个局部应用,然后以这一层次的数据流图为出发点,设计分 E-R 图。
通常以中层数据流图作为设计分 E-R 图的依据。原因:
1.高层数据流图只能反映系统的概貌
2.中层数据流图能较好地反映系统中各局部应用的子系统组成
3.低层数据流图过细

逐一设计分 **E-R** 图

任务:标定局部应用中的实体、属性、码,实体间的联系

如何抽象实体和属性

实体:现实世界中一组具有某些共同特性和行为的对象就可以抽象为一个实体。对象和实体之间是“ is member of" 的关系。
属性:对象类型的组成成分可以抽象为实体的属性。组成成分与对象类型之间是“ is part of" 的关系。

如何区分实体和属性

实体与属性是相对而言的。同一事物,在一种应用环境中作为“属性”,在另一种应用环境中就必须作为“实体”。
属性不能再具有需要描述的性质。即属性必须是不可分的数据项,不能再由另一些属性组成。
属性不能与其他实体具有联系。联系只发生在实体之间。
符合上述两条特性的事物一般作为属性对待。
为了简化 E-R 图的处置,现实世界中的事物凡能够作为属性对待的,应尽量作为属性。

步骤

1．以数据字典为出发点定义 E-R 图。
数据字典中的“数据结构”、“数据流”和“数据存储”等已是若干属性的有意义的聚
合
2．按上面给出的准则进行必要的调整。

视图的集成

各个局部视图即分 E-R 图建立好后,还需要对它们进行合并,集成为一个整体
的数据概念结构即总 E-R 图。

方式

1.一次集成(通常用于局部视图比较简单时)
2.逐步累积式
首先集成两个局部视图(通常是比较关键的两个局部视图)
以后每次将一个新的局部视图集成进来

步骤

1.合并
2.修改与重构

冲突的种类

属性冲突
命名冲突
结构冲突

两类属性冲突

属性域冲突:属性值的类型、取值范围或取值集合不同。
属性取值单位冲突

命名冲突

同名异义:不同意义的对象在不同的局部应用中
异名同义(一义多名):同一意义的对象在不同的局部应用中具有不同的名字具有相同的名字
命名冲突可能发生在属性级、实体级、联系级上。其中属性的命名冲突更为常
见。

结构冲突

１．同一对象在不同应用中具有不同的抽象
解决方法:通常是把属性变换为实体或把实体变换为属性,使同一对象具有相同的
抽象。变换时要遵循两个准则。
２．同一实体在不同局部视图中所包含的属性不完全相同,或者属性的排列次序不完全相同。
产生原因:不同的局部应用关心的是该实体的不同侧面。
解决方法:使该实体的属性取各分 E-R 图中属性的并集,再适当设计属性的次序 。
3．实体之间的联系在不同局部视图中呈现不同的类型
解决方法:根据应用语义对实体联系的类型进行综合或调整。

冗余

冗余的数据是指可由基本数据导出的数据,冗余的联系是指可由其他联系导出的联系。
冗余数据和冗余联系容易破坏数据库的完整性,给数据库维护增加困难
并不是所有的冗余数据与冗余联系都必须加以消除,有时为了提高某些应用的效率,不得不以冗余信息作为代价。
设计数据库概念结构时,哪些冗余信息必须消除,哪些冗余信息允许存在,需要根据用户的整体需求来确定。
消除不必要的冗余后的初步 E-R 图称为基本 E-R 图。

消除冗余的方法

1.分析方法
以数据字典和数据流图为依据,根据数据字典中关于数据项之间逻辑关系的说明来消除冗余。
2.规范化理论
函数依赖的概念提供了消除冗余联系的形式化工具

逻辑结构设计

任务

概念结构是各种数据模型的共同基础
为了能够用某一 DBMS 实现用户需求,还必须将概念结构进一步转化为相应的数据模型,这正是数据库逻辑结构设计所要完成的任务。

步骤

将概念结构转化为一般的关系、网状、层次模型
将转化来的关系、网状、层次模型向特定DBMS 支持下的数据模型转换
对数据模型进行优化

E-R 图向关系模型的转换

1.转换内容
E-R 图由实体、实体的属性和实体之间的联系三个要素组成
关系模型的逻辑结构是一组关系模式的集合
将 E-R 图转换为关系模型:将实体、实体的属性和实体之间的联系转化为关系模式。
2.转换原则
1一个实体型转换为一个关系模式。(关系的属性:实体型的属性;关系的码:实体型的码)
2一个 m:n 联系转换为一个关系模式。(关系的属性:与该联系相连的各实体的码以及联系本身的属性;关系的码:各实体码的组合)
3一个 1:n 联系可以转换为一个独立的关系模式,也可以与 n 端对应的关系模式合并。
1.转换为一个独立的关系模式(关系的属性:与该联系相连的各实体的码以及联系本身的属性;关系的码: n 端实体的码)
2.与 n 端对应的关系模式合并(合并后关系的属性:在 n 端关系中加入 1 端关系的码和联系本身的属性;合并后关系的码:不变)
2可以减少系统中的关系个数，一般情况下更倾向于采用这种方法
４.一个 1:1 联系可以转换为一个独立的关系模式,也可以与任意一端对应的关系模式合并。
1.转换为一个独立的关系模式(关系的属性:与该联系相连的各实体的码以及联系本身的属性;关系的候选码:每个实体的码均是该关系候选码)
2.与某一端对应的关系模式合并(合并后关系的属性:加入对应关系的码和联系本身的属性;合并后关系的码:不变)
5.三个或三个以上实体间的一个多元联系转换为一个关系模式。(关系的属性:与该多元联系相连的各实体的码以及联系本身的属性;关系的码:各实体码的组合)
6.同一实体集的实体间的联系,即自联系,也可按上述 1:1 、 1:n 和 m:n 三种情况分别处理。

7.具有相同码的关系模式可合并。(目的:减少系统中的关系个数。合并方法:将其中一个关系模式的全部属性加入到另一个关系模式中,然后去掉其中的同义属性(可能同名也可能不同名),并适当调整属性的次序。)

向特定 DBMS 规定的模型进行转换

一般的数据模型还需要向特定 DBMS 规定的模型进行转换。转换的主要依据是所选用的 DBMS 的功能及限制。没有通用规则。对于关系模型来说,这种转换通常都比较简单。

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,这就是数据模型的
优化。关系数据模型的优化通常以规范化理论为指导。

方法

1.确定数据依赖
按需求分析阶段所得到的语义,分别写出每个关系模式内部各属性之间的数据依赖
以及不同关系模式属性之间数据依赖 。
2.对于各个关系模式之间的数据依赖进行极小化处理,消除冗余的联系。
3.按照数据依赖的理论对关系模式逐一进行分析,考查是否存在部分函数依赖、传递函数依赖、多值依赖等,确定各关系模式分别属于第几范式。
4.按照需求分析阶段得到的各种应用对数据处理的要求,分析对于这样的应用环境这些模式是否合适,确定是否要对它们进行合并或分解。
5.按照需求分析阶段得到的各种应用对数据处理的要求,对关系模式进行必要的分解或合并,以提高数据操作的效率和存储空间的利用率

常用分解方法

1.水平分解
把 (基本) 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,以提高系统的效率。
适用范围：１． 满足“ 80/20 原则”的应用2. 并发事务经常存取不相交的数据
2.垂直分解
把关系模式 R 的属性分解为若干子集合,形成若干子关系模式。
原则：经常在一起使用的属性从 R 中分解出来形成一个子关系模式。
优点：可以提高某些事务的效率
缺点：可能使另一些事务不得不执行连接操作,从而降低了效率。
适用范围：取决于分解后 R 上的所有事务的总效率是否得到了提高。
方法：简单情况:直观分解;复杂情况:用第五章中的模式分解算法;垂直分解必须不损失关系模式的语义 (保持无损连接性和保持函数依赖) 。

用户子模式

定义用户外模式时应该更注重考虑用户的习惯与方便。包括三个方面:
1.使用更符合用户习惯的别名
2.针对不同级别的用户定义不同的外模式,以满足系统对安全性的要求。
3.简化用户对系统的使用

数据库的物理设计

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,它依赖于选定的数据库管理系统。为一个给定的逻辑数据模型选取一个最适合应用要求的物理结构的过程,就是数据库的物理设计。

步骤

1.确定数据库的物理结构
2.对物理结构进行评价,评价的重点是时间和空间效率
3.如果评价结果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。

聚簇

为了提高某个属性(或属性组)方法的查询速度,把这个或这些属性(称为聚簇码)方法上具有相同值的元组集中存放在连续的物理块称为聚簇,许多关系型 DBMS 都提供了聚簇功能
建立聚簇索引后,基表中数据也需要按指定的聚簇属性值的升序或降序存放。也即聚簇索引的索引项顺序与表中元组的物理顺序一致。
用途:对于某些类型的查询,可以提高查询效率
适用范围:很少对基表进行增删操作,很少对其中的变长列进行修改操作
优点：1，大大提高按聚簇属性进行查询的效率，2，节省存储空间
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

HASH

当一个关系满足下列两个条件时,可以选择
HASH 存取方法:
1.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
2.该关系的大小可预知,而且不变;或该关系的大小动态改变,但所选用的 DBMS 提供了动态 HASH 存取方法。

数据存放位置

影响数据存放位置和存储结构的因素:1,硬件环境;2,应用需求(存取时间;存储空间利用率;维护代价)
根据应用情况将(易变部分与稳定部分),(存取频率较高部分与存取频率较低部分)分开存放,以提高系统性能

物理结构的评价

内容：对数据库物理设计过程中产生的多种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
方法：
1.定量估算各种方案(存储空间;存取时间;维护代价;)
2.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
3.如果该结构不符合用户需求,则需要修改设计

数据库实施

内容：
１．用 DDL 定义数据库结构；
２．组织数据入库;
３．编制与调试应用程序;
４．数据库试运行

数据装载

数据库结构建立好后,就可以向数据库中装载数据了。组织数据入库是数据库实施阶段最主要的工作。
方法：1．人工方法；2，计算机辅助数据入库

人工方法

适用于小型系统
１．筛选数据
２．转换数据格式
３．输入数据
４．校验数据

计算机辅助数据入库

１．筛选数据
２．输入数据
３．校验数据
４．转换数据
５．综合数据

运行和维护

数据库的转储和恢复
数据库的安全性、完整性控制
数据库性能的监督、分析和改进
数据库的重组织和重构造