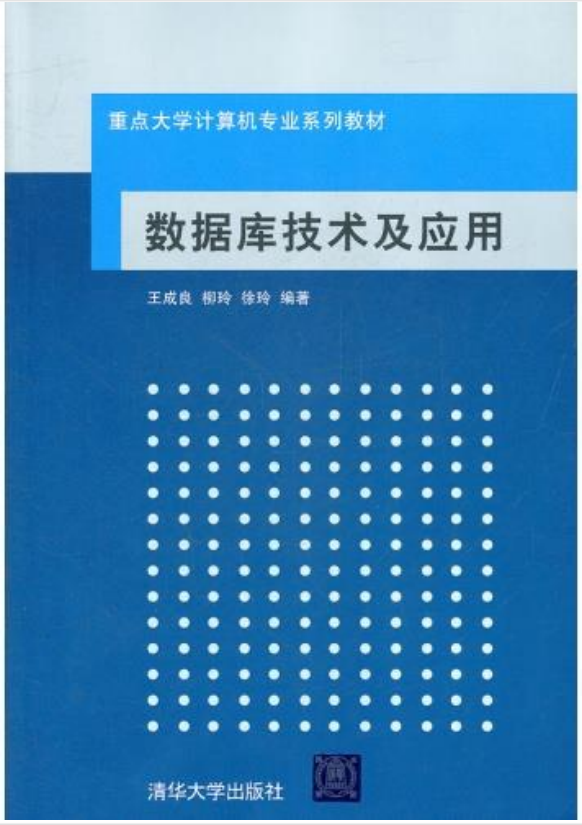
数据库原理

柳玲 重庆大学软件学院

LLing29@tom.com

参考资料

- 王成良，柳玲，徐玲. 数据库技术及应用. 清华大学出版社. 2011 年11月



主要内容

- 第1章 绪论
- 第2章 关系模型
- 第3章 结构化查询语言－SQL
- 第4章 关系数据库的规范化设计
- 第5章 数据库设计
- 第6章 事务

第1章 绪论

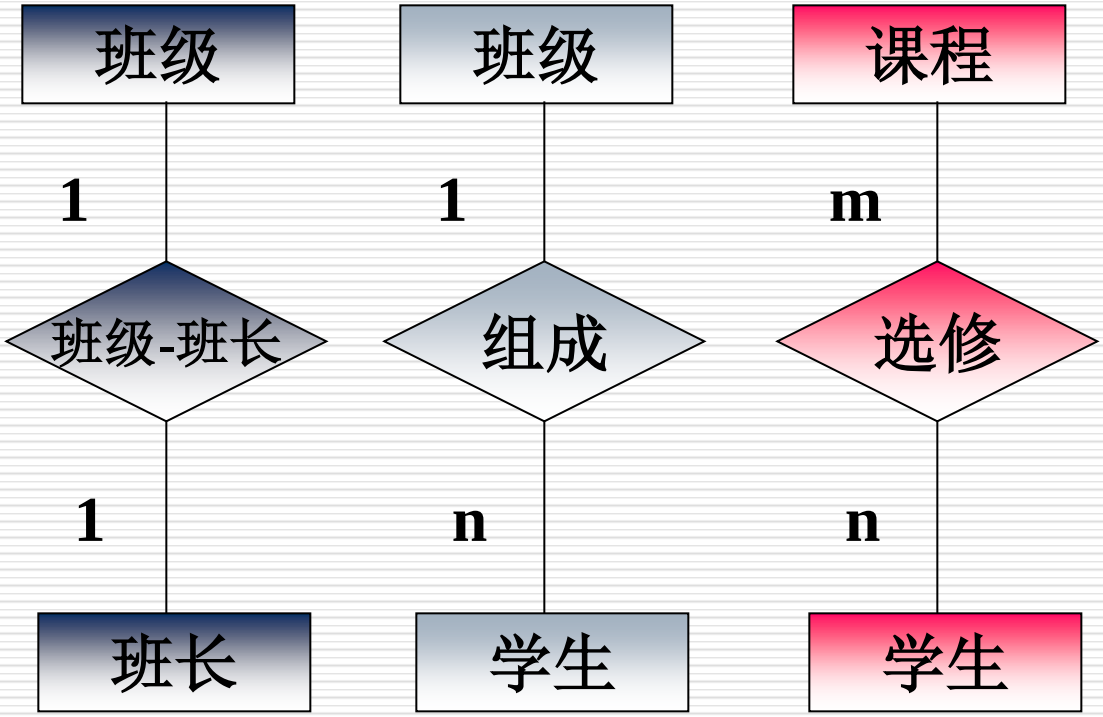
- 基本概念
  - 数据库（Database，DB）：长期储存在计算机内、有组织的、可共享的数据集合。
- 数据库特点
  - 数据的结构化
  - 数据的冗余度小
  - 较高的数据独立性。

- 数据库管理系统（Database Management System，DBMS）：是位于用户与操作系统之间的一层数据管理软件。
- 一般来说，DBMS的功能主要包括以下6个方面：
- 1）数据库的定义功能。2）数据库的操纵功能。3）数据库的保护功能。4）数据库的存储管理。5）数据库的维护功能。6）数据字典。

第1章 绪论

E-R 图

- 实体：客观存在并可相互区别的事物。
- 联系：
  - 一对一联系 (1：1)
  - 一对多联系 (1：N)
  - 多对多联系 (M：N)



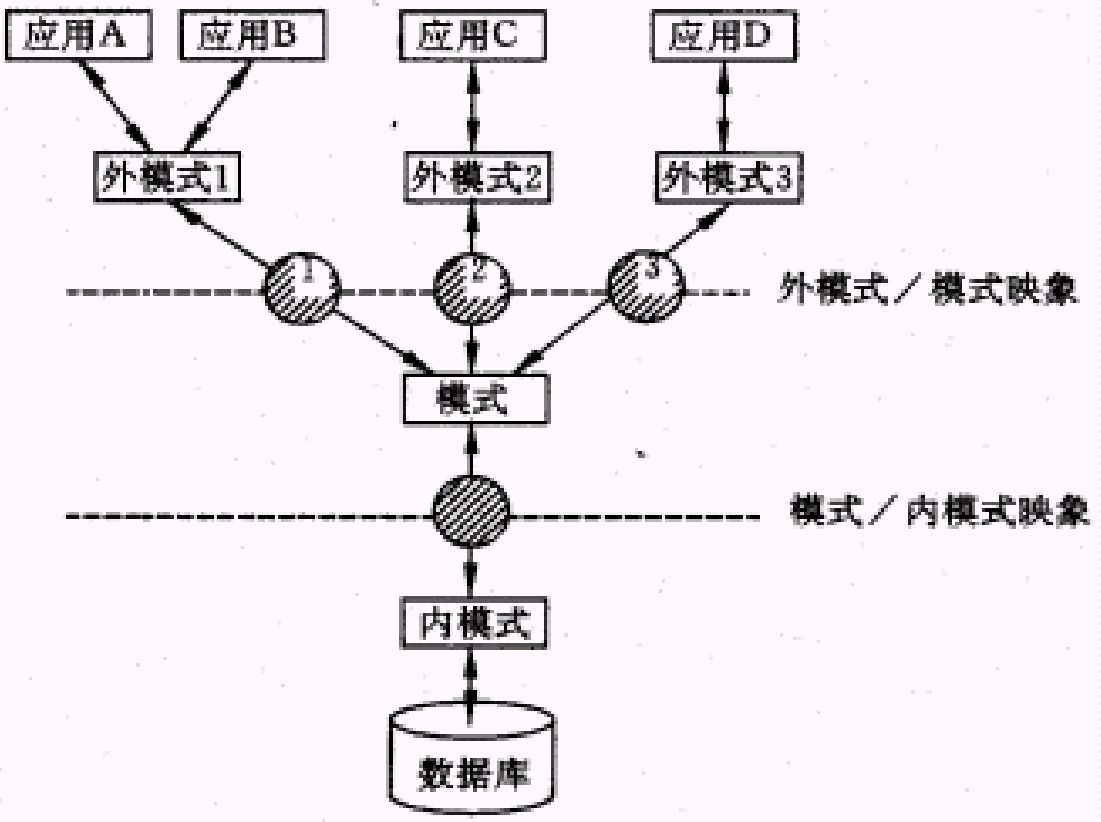
第1章 绪论

- E-R模型向关系模型转换的规则是：
- 1）实体：一个实体类型转换成一个关系模式。实体的属性就是关系的属性，实体的码就是关系的码。
- 2） 一对一（1：1）联系
- 一般是将联系与任意一端实体所对应的关系模式合并，需要在该关系模式的属性中加入另一个实体的码和联系本身的属性。
- 3） 一对多（1:n）联系
- 一般是将该联系与n端实体所对应的关系模式合并。合并时需要在n端实体的关系模式的属性中加入1端实体的码和联系本身的属性。
- 4）多对多(m:n)联系
- 将联系转换成一个关系模式。与该联系相连的各实体的码及联系本身的属性转换为关系的属性，而关系的码为各实体码的组合。

第1章 绪论

- 数据模型：是现实世界数据特征的抽象。
- 概念数据模型：按用户的观点对数据和信息建模。如：实体、联系模型
- 逻辑数据模型：按计算机系统的观点对数据建模。如：层次模型，网状模型，关系模型

第1章 绪论



数据库系统的模式结构

- 模式（概念模式）是数据库中全体数据的逻辑结构和特征的描述。
- 外模式（子模式或用户模式），是数据库用户使用的局部数据的逻辑结构和特征的描述。
- 内模式（存储模式）是数据物理结构和存储结构的描述。
- 数据的逻辑独立性
- 数据的物理独立性

第2章 关系模型

- 超键（Superkey）：在一个关系中，能唯一标识元组的属性集。
- 键Key(候选键candidate key)：一个属性集能惟一标识元组，又不含有多余属性。
- 主键（primary key）：关系模式中用户正在使用的候选键称主键。
- 主属性（prime attribute）：主键的诸属性。

第2章 关系模型

- 外键（foreign key）：设F是基本关系R的一个或一组属性，但不是R的键，如果F与基本关系S的主键Ks相对映，则称F是R的外键。
- 关系的完整性
  - 实体完整性：如属性A是基本关系R的主属性，则属性A不能取空值。
  - 参照完整性规则：外键值必须为：
    - 或者取空值（F的每个属性值均为空值）；
    - 或者等于S中某个元组的主键值。
  - 用户定义的完整性：针对某一个具体关系数据库的约束条件，反映某一具体应用所涉及的数据必须满足的语义要求。

第2章 关系模型

- 关系代数：用关系的运算来表达查询要求的方式。
- 集合运算
- 设关系R和关系S具有相同的n个属性，且相应的属性取自同一个域，
- （1）并
- $R \cup S=\{t \mid t \in R \vee t \in S\}$
- （2） 差
- $R-S=\{t \mid t \in R \wedge t \overline{\in} S\}$
- （3）交
- $R \cap S=\{t \mid t \in R \wedge t \in S\}$

第2章 关系模型

- (4) 广义笛卡尔积
- 两个分别为n目和m目的关系R和S的广义笛卡尔积是一个(n+m)列的元组的集合。元组的前n列是关系R的一个元组，后m列是关系S的一个元组。若R有k1个元组，S有k2个元组，则关系R和关系S的广义笛卡尔积有k1×k2个元组。
- 设关系R和S的属性个数分别为r和s，则(R×S)操作结果的属性个数为（　　）
- A、r+s　　B、r-s　　C、r×s　　D、max(r,s)

第2章 关系模型

- 投影：是从关系R中选择出若干属性列组成新的关系。
- 设关系 R是 k元关系，R在其分量 Ai1，…，Aim上的投影用 $\pi_{i1,\ldots,im}(R)$ 表示：
- $\pi_{i1,\ldots,im}(R) = \{t \mid t=\langle t_{i1},\ldots,t_{im}\rangle \wedge \langle t_1,\ldots,t_k\rangle \in R\}$
- 选择：在关系R中选择满足给定条件的元组。
- $\sigma_F(R) = \{t \mid t \in R \wedge F(t) = \text{true}\}$
- 例如，$\sigma_{2>'3'}(R)$ 表示从R中挑选第2个分量值大于3的元组所构成的关系。
- 连接：从两个关系的笛卡儿积中选取属性间满足一定条件的元组。

关系代数运算的应用实例

- 设教学数据库中有三个关系：
- 学生关系 S（S#，SNAME，AGE，SEX）
- 选课关系 SC（S#，C#，GRADE）
- 课程关系 C（C#，CNAME，TEACHER）
- 用关系代数表达式表达每个查询语句。

（1）检索学习课程号为C2的学生学号与成绩。

$$\pi_{S\#,\ GRADE}\left(\sigma_{C\#='C2'}(SC)\right)$$

（2）检索选修课程名为MATHS的学生学号与姓名。

$$\pi_{S\#,\ SNAME}\left(\sigma_{CNAME='MATHS'}(S\bowtie SC\bowtie C)\right)$$

（3）检索不学C2课的学生姓名与年龄。

$$\pi_{SNAME,\ AGE}(S)-\pi_{SNAME,\ AGE}\left(\sigma_{C\#='C2'}(S\bowtie SC)\right)$$

第3章　结构化查询语言－SQL

- 1.定义基本表
- CREATE TABLE〈表名〉(〈列名〉〈数据类型〉[列级完整性约束条件][,〈列名〉〈数据类型〉[列级完整性约束条件]...)[,〈表级完整性约束条件〉];

- 删除基本表
- DROP　TABLE〈表名〉

```
CREATE TABLE SC
(SNO CHAR(5) NOT NULL
CONSTRAINT S_FORE FOREIGN KEY
REFERENCES S(SNO),
CNO CHAR(5) NOT NULL
CONSTRAINT C_FORE FOREIGN KEY
REFERENCES C(CNO),
SCORE NUMERIC(3),
CONSTRAINT S_C_PRIM PRIMARY
KEY (SNO,CNO));
```

第3章　结构化查询语言－SQL

- 索引是对数据库表中一列或多列的值进行排序的一种结构。
- 索引提高了查询的速度,但一般会降低更新的速度。
- CREATE [UNIQUE] [CLUSTER] INDEX 〈索引名〉 ON 〈表名〉 (〈列名〉[〈次序〉] [,〈列名〉[〈次序〉]]...);
- 索引可以建在表的一列或多列上。可在每个〈列名〉后面指定索引值的排列次序。ASC表示升序，DESC表示降序，缺省值为ASC。
- UNIQUE表明建唯一性索引。
- CLUSTER表示建聚簇索引。所谓聚簇索引是指索引项的顺序与表中记录的物理顺序一致的索引。

第3章　结构化查询语言－SQL

- 查询
- SELECT [ALL|DISTINCT]<目标列表达式>[,<目标列表达式>]... FROM <表名或视图名>[,<表名或视图名>] ... [WHERE <条件表达式>] [GROUP BY <列名1>[HAVING <条件表达式>]] [ORDER BY <列名2> [ASC|DESC]];
- GROUP：将结果按<列名1>的值进行分组，该属性列值相等的元组为一个组，每个组产生结果表中的一条记录。
- 如果GROUP子句带HAVING短语，则只有满足指定条件的组才予输出。
- 如果有ORDER子句，则结果表还要按<列名2>的值的升序或降序排序。

第3章　结构化查询语言－SQL

- 使用聚合函数
- COUNT([DISTINCT|ALL] *) 统计元组个数
- COUNT([DISTINCT|ALL] 〈列名〉) 统计一列中值的个数
- SUM([DISTINCT|ALL] 〈列名〉) 计算一列值的总和
- AVG([DISTINCT|ALL] 〈列名〉) 计算一列值的平均值
- MAX([DISTINCT|ALL] 〈列名〉) 求一列值中的最大值
- MIN([DISTINCT|ALL] 〈列名〉) 求一列值中的最小值

在数据库jxsk中，包含下列五张表：

学生表S

sno	sn	sex	age	dept
s2	钱尔	男	18	信息
s5	周武	男	20	计算机

选课表SC

sno	cno	score
s2	c5	57
s3	c1	75

课程表C

cno	cn	ct	id_tc
c1	程序设计	60	1
c2	微机原理	80	2

教师表T

tno	tn	sex	age	prof	sal	comm	dept
t5	张兰	女	39	副教授	1300	2000	信息
t4	张雪	女	51	教授	1600	3000	自动化

授课表TC

tno	cno
t2	c5
t3	c1

- 查询选修了3门以上课程的学生姓名和平均成绩.
- Select sn as 姓名,avg(score) as 平均成绩 from s sc where s.sno=sc.sno group by s.sno having count(*)>3

第3章　结构化查询语言－SQL

- 视图是一个虚拟表，其内容由查询定义。同真实的表一样，视图包含一系列带有名称的列和行数据。但是，视图并不在数据库中以存储的数据值集形式存在。行和列数据来自由定义视图的查询所引用的表，并且在引用视图时动态生成。
- 优点：
- 简化操作：在定义视图时，若视图本身就是一个复杂查询的结果集，向用户隐藏了表与表之间的复杂的连接操作。
- 安全性：通过视图，用户只能查看和修改他们所能看到的数据，其它数据库或表既不可见也不可以访问。
- 逻辑数据独立性。视图可帮助用户屏蔽真实表结构变化带来的影响。

第3章　结构化查询语言－SQL

- 谓词LIKE可用来进行字符串的匹配。语法格式：
  - [NOT] LIKE ’<匹配串>’ [ESCAPE ’<换码字符>’]
- 含义是查找指定的属性列值与<匹配串>相匹配的元组。
- %(百分号) 代表任意长度（长度可以为0）的字符串。
- _(下横线) 代表任意单个字符。

第3章　结构化查询语言－SQL

- 例：下面哪一个语句能查找名称以“book”字符串结尾的出版社？
- A. Select pub_name from publishers where pub_name like ‘_book’
- B. Select pub_name from publishers where pub_name like ‘%book’
- C. Select pub_name from publishers where pub_name like ‘^book ’
- D. Select pub_name from publishers where pub_name like ‘[book’

第4章 关系数据库的规范化设计

- **函数依赖**：设R(U)是属性集U上的关系模式，X，Y ⊆ U，r是R(U)上的任意一个关系，如果成立对∀t 1，t2 ∈ r，若t1[X] = t2[X]，则t1[Y] = t2[Y]，那么称“X函数决定Y”，或“Y函数依赖于X”，记作X→Y。称X为决定因素。

  如S# → Sname，　(S#，C#) → Grade

第4章 关系数据库的规范化设计

- 关系的1NF性质要求元组分量不可再分解。即不能以集合、序列等作为属性值。

- 设X->Y是关系模式R的一个函数依赖，如果存在X的真子集X‘,使得X’->Y成立，则称Y部分依赖于X。

- 如果关系模式R中的所有的非主属性都不部分依赖于任何一个候选关键字，则称关系R是属于第二范式。

第4章 关系数据库的规范化设计

- 如果关系模式R中的所有的非主属性都不传递依赖于任何一个候选关键字，则称关系R是属于第三范式（3NF，Third Normal Form）。

- 例：一个关系模式为Y(X1,X2,X3,X4)，假定该关系存在着如下函数依赖：(X1,X2)->X3,X1->X4,则该关系属于第一范式，因它存在着非主属性部分函数依赖于候选键。

第6章　并发控制

- 日志文件是用来记录事务对数据库的更新操作的文件，由系统自动记录。不同数据库采用的日志文件格式不一样。
- 设立日志文件的意义：事务故障恢复和系统故障恢复必须用日志；在动态转储方式中必须转存日志文件，由后援副本和日志结合有效恢复数据库；在静态转储方式中，利用日志文件可以恢复到故障前某一时刻的正确状态。

第5章　数据库设计

- 数据库设计包括六个主要步骤：
  - 需求分析：了解用户的数据需求、处理需求、安全性及完整性要求；
  - 概念设计：通过数据抽象，设计系统概念模型，一般为E-R模型；
  - 逻辑结构设计：设计系统的模式和外模式，对于关系模型主要是基本表和视图；
  - 物理结构设计：设计数据的存储结构和存取方法，如索引的设计；
  - 系统实施：组织数据入库、编制应用程序、试运行；
  - 运行维护：系统投入运行，长期的维护工作。

第6章　并发控制

- 事务是由一系列操作序列构成的程序执行单元。
- 事务的四个特性：ACID
- 原子性(Atomicity)：事务中包含的所有操作要么全做，要么全不做。
- 一致性(Consistency)：事务的隔离执行必须保证数据库的一致性。事务开始前，数据库处于一致性的状态；事务结束后，数据库必须仍处于一致性状态。
- 隔离性(Isolation)：系统必须保证事务不受其它并发执行事务的影响。对任何一对事务T1，T2，在T1看来，T2要么在T1开始之前已经结束，要么在T1完成之后再开始执行。
- 持久性(Durability)：一个事务一旦提交之后，它对数据库的影响必须是永久的，即使系统出现故障时也如此。

谢谢!