系统分析师（数据库系统）

发件人：cyg_mail<cyg_mail@163.com>
时　间：2015年4月2日(星期四) 上午10:16
收件人：光仔<494637923@qq.com>

第五章　数据库系统

必考：规范化理论（函数依赖、候选关键字、范式、模式分解的概念及方法，元组演算）。
其他重点：数据库设计（ERP模型）、数据仓库（数据仓库的概念和分类）。

数据库会出现在上午、下午的试题当中，特别是数据库安全、分布式数据和数据挖掘。

数据库系统考题分数分布

知识点	2004.5	2004.11	2005.5	2005.11	2006.5	合计	重点程度
关系运算		2				2	
范式理论	2	2	3	1	4	12	★★★★★
并发控制	1					1	
数据库设计		1		4		5	★★★
数据库安全			1			1	
分布式数据库	1		1			2	
数据仓库	3		1			4	★★
数据挖掘					1	1	
合计	7	5	6	5	5	28	

数据库模式（三级模式+两级映射）

数据库系统是指在信息系统中引入数据库后的系统，一般由数据库、数据库管理系统（DBMS）、应用系统和数据库管理员（DMA）、用户组成。
数据库结构典型划分有三级模式和两级映射。

三级模式

外模式

也成为子模式、用户模式，对应于用户级数据库。用于描述用户看到的那部分数据的逻辑结构（用户视图）
一个数据库可以有多个外模式，一个应用程序只能使用一个外模式。

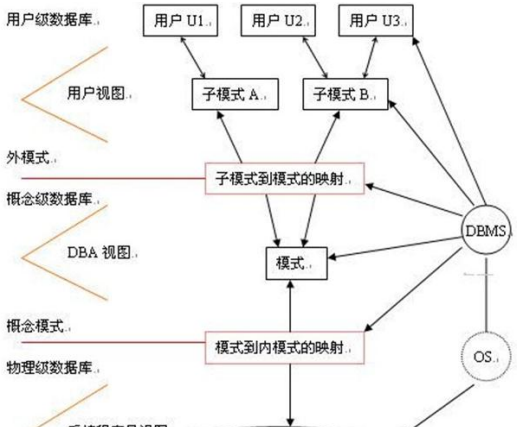
概念模式

也成为模式、逻辑模式，对应于概念级数据库。描述数据的逻辑结构和特征，实体联系、数据定义、完整性约束等。此外还包括访问控制、保密和完整性约束等方面的检查。
一个数据库只有一个概念模式。

内模式

对应于物理级数据库，是数据物理结构和存储方式的描述，是数据在数据库内部的表示方式。是存储记录、索引和存储路径等数据的存储组织。
一个数据库只有一个内模式。

数据库的三级模式中，模式是数据库的中心和关键。
内模式依赖于模式，独立于外模式和存储设备；
外模式面向具体应用，独立于模式和存储设备；
应用程序则依赖于外模式，独立于模式和内模式。



两级独立性（两级映射）

数据库系统的两级独立性指物理独立性和逻辑独立性。

三个抽象级别通过**两级映射（外模式<=>模式，模式<=>内模式）**进行相互转换，使得三级模式形成一个独立的整体。

物理独立性存在于概念模式和内模式之间的映射转换。即物理位置改变，应用程序不用改变。

逻辑独立性存在于外模式和概念模式之间的映射转换。即数据逻辑结构改变，应用程序不用改变。

数据模型

数据模型三要素：

数据结构：描述数据类型、内容、性质和数据间的联系，是数据模型的基础，数据操作和数据约束都建立在此基础之上。

数据操作：定义数据结构上的操作类型和操作方式；

数据约束：描述数据结构内数据的语法和语意联系，他们之间的约束和依存关系，数据的动态变化规则，以保证数据的正确、有效和相容。

数据模型分类

数据模型主要分类两大类：概念数据模型（实体联系模式）、基本数据模型（结构数据模型）；

概念数据模型

按照用户的观点来对数据和信息建模，主要用于设局库的涉及，一般用**实体-联系（Entity-Relationship，E-R）**方法表示。

基本数据模型

按照计算机系统的观点对数据和信息建模，主要用于数据库的实现。其中著名的基本数据模型有：

层次模型：最早的数据模型，采用树状结构作为组织方式，典型是IBM的IMS系统，现已被淘汰。

网状模型：使用有向图表示实体类型和实体之间的联系，记录之间通过指针实现，效率高但编程复杂，必须熟知数据库逻辑结构，层次模型是网状模型的特列。

关系模型：用表格表示实体集，外键表示实体间联系，存储路径透明，能简化开发工作，但效率不高。关系模型是最广泛使用的一种模型，使用关系模型的数据库系统，称为关系数据库系统。

面向对象模型：用面向对象的观点来描述现实世界实体的逻辑组织、对象之间的限制和联系。目前应用并不多。

关系模型

关系模式通常简记为R（a1,a2,...,an），其中R为关系名，a1,a2,...,an为属性名。

关系实际上就是关系模式在某一时刻的状态或内容，也就是说关系模式是型，关系是它的值。关系模式是静态的，稳定的，关系是动态的，随时间不短变化的。

实际应用中，将关系模式和关系统称为关系。

关系运算

并

给出关系R,S（具有相同列数），$R \cup S$ 包括R,S的所有元组，定义为：$R \cup S = \{t \mid t \in R \vee t \in S\}$

差

给出关系R,S（具有相同列数），$R - S$ 包括R中有，而S中没有的元组的集合，定义为：$R \cup S = \{t \mid t \in R \vee t \notin S\}$

交

给出关系R,S（具有相同列数），$R \cap S$ 包括R和S中相同元组的集合，定义为：$R \cap S = \{t \mid t \in R \wedge t \in S\}$

笛卡尔积

令关系R为m元的关系，S为n元的关系，则 $R \times S$ 是m+n元的元组的集合，关系的前m个元素来自R的一个元组，后n个元素来自S的一个元组，定义：$R \times S \equiv \{t \mid t = < t_r, t_s > \wedge t_r \in R \wedge t_s \in S\}$，若R有u个元组，S有v个元组，则RXS有u*v个元组。

投影

从一个关系中抽取指明的属性（列），令R为一个包含A属性的关系，$\Pi_A(R) \equiv \{t[A] \mid t \in R\}$（SQL中的SELECT吗？）

选择

从关系R中抽取出满足给定限制条件的记录，记作：$\sigma_F(R) \equiv \{t \mid t \in R \wedge F(t) = true\}$（SQL中的WHERE吗？）

θ 连接

θ连接从两个关系的笛卡尔积中选取属性之间满足一定条件的元组。记作：$R \underset{A\theta B}{\triangleright\triangleleft} S \equiv \{t_r t_s \mid t_r \in R \wedge t_s \in S \wedge t_r[A] \theta t_s[B]\}$,A,B为R和S上元数相等且可比的

$R \triangleright\triangleleft S \equiv \{t_r t_s \mid t_r \in R \wedge t_s \in S \wedge t_r[A] = t_s[B]\}$

属性组。Θ为等号时，称为自然连接：

除

设关系R(X,Y)与关系S(Z),Y和Z具有相同的属性个数，其对应属性出自相同域。关系 $R(X,Y)\div S(Z)$,所得的商是关系R在属性X上的投影的一个子集，该子集和S(Z)的笛卡尔积必须包含在R(X,Y)中，记为 $R\div S$.计算公式略。简单地说，就是求R中的Y属性与S中的Z属性相同的X属性。

元组演算（关系演算）

http://zh.wikipedia.org/wiki/%E5%85%83%E7%BB%84%E5%85%B3%E7%B3%BB%E6%BC%94%E7%AE%97

一般形式{[t|P(t)]},其中t是元组变量，P是一个公式（谓词，表达式），{[t|P(t)]}表示满足关系P的算有元组。

三种原子公式

R(s)：R是关系名，s是元组变量；

s[i]Θu[j]：s和u是元组变量，Θ是算术运算符

s[i]Θa或aΘu[j]：其中a为常量。

公式：

(1) 每个原子是一个公式。其中的元组变量是自由变量。
(2) 如果 P1 和 P2 是公式，那么 ¬P1、P1∨P2、P1∧P2 和 P1→P2 也是公式。
(3) 如果 P1 是公式，那么 (∃s)(P1)和(∀s)(P1)也都是公式。
(4) 公式中各种运算符的优先级从高到低依次为：θ，∃和∀，¬，∧和∨，→。

除以上之外都不是公式。

转换规则

(1) P1∧P2 等价于 ¬(¬P1∨¬P2)。
(2) P1∨P2 等价于 ¬(¬P1∧¬P2)。
(3) (∀s)(P1(s))等价于 ¬(∃s)(¬P1(s))。
(∃s)(P1(s))等价于 ¬(∀s)(¬P1(s))。
(4) P1→P2 等价于 ¬P1∨P2。

关系代数表达式转换为元组表达式

(1) R∪S 可用 [t|R(t)∨S(t)]表示。
(2) R-S 可用 [t|R(t)∧¬S(t)] 表示。
(3) R×S 可用 [t|(∃u)(∃v)(R(u)∧S(v)∧t[1]=u[1]∧t[r]=u[r]∧…∧t[r]=u[r]∧t[r+1]=v[1]∧…∧t[r+s]=v[s])] 表示，此处设 R 是 r 元，S 是 s 元。
(4) 设投影操作是 $\pi_{2,3}(R)$，那么元组表达式可写成 [t|(∃u)(R(u)∧t[1]=u[2]∧t[2]=u[3])]。
(5) $\sigma_F(R)$ 可用 [t|R(t)∧F']表示，F'是 F 的等价表示形式。例如，$\sigma_{2='d'}(R)$ 可写成 [t|(R(t)∧t[2]='d')。

（图中RxS的表示方法错误）

规范化理论(范式)

又称数据库标准化。

函数依赖与键

键决可定其他属性，或者其他属性依赖于键。若属性Y完全依赖于键X，则称Y对X完全函数依赖，或者X决定Y。记作：$X \to Y$

关系模式的键也称作码或关键字。

第一范式（1NF）

数据库的每个字段都只能存放单一值，而且每笔记录都要能利用一个惟一的主键来加以识别。

第二范式（1NF）

第三范式（1NF）

BCNF

第四范式（1NF）

数据库访问接口

数据库接口是指应用程序与数据库之间的连接部分。主要的分类：

1、专用调用：

2、开放数据库互连（Open DataBase Connectivity,ODBC）：

3、JAVA数据库连接(Java DataBase Connectivity,JDBC)：

数据库的控制功能

并发控制：许多**事务**可能同时对同一数据进行操作，称为并发操作。DMBS的并发控制子系统负责协调并发事务的执行，保证数据库的完整性不受破坏。避免用户得到不正确的数据。

事务

事务是用户定义的一个数据库操作序列，是一个不可分割的工作单位（要么全做，要么不做）。数据库事务具有以下特定：

原子性：保证事务包含的一组更新操作是不可分的一个整体。

一致性：指数据库从一个一致性状态变到另一个一致性状态，如账户金额。

隔离性：一个事物的执行不能被其他事务干扰，要求即使多个事务并发执行，但看上去像串行调度执行一样，也称可串行性。

持久性：指事务一旦提交，无论发生何种故障，改变都是持久的。

将数据库事务的特性统称为**ACID**特性。

数据不一致问题

修改丢失：事务A与事务B从数据库中同时读入一数据并修改，结果一个事务提交的结果破坏了对另一个事务提交的结果。

读"脏数据"：事务A修改一处数据，写回磁盘，事务B读取同一数据后，事务A撤销该操作（将该数据恢复原值），此时事务B读取的数据不一致。

不可重复读：事务A读取一数据，事务B执行更新操作，使得事务A使用的是更新前的值。

封锁协议

处理并发控制主要采用封锁技术，主要有X封锁和S封锁。

（1）排他型封锁（X封锁）：事务独锁某个数据，具有排他性。

（2）共享型封锁（S封锁）：允许并发读，不允许修改。

三级封锁协议

一级封锁：事务T在修改数据R之前必须先对其加X锁，直到事务结束才释放。可以**防止**修改丢失，但**不能防止**读脏数据和不可重复读。

二级封锁：一级封锁协议，加上事务T在读取数据R之前先对其加S锁，读完后立即释放S锁。可以**防止**修改丢失、读脏数据，但**不能防止**不可重复读。

三级封锁：一级封锁协议，加上事务T在读取数据R之前先对其加S锁，直至事务结束才释放。可以**防止**修改丢失、读脏数据、不可重复读。

两段封锁协议：事务分两个阶段来对数据项加锁和解锁。其中**扩展阶段**在对任何数据进行读写之前，首先要申请并获得对该数据的封锁；**收缩阶段**是在释放一个锁后，事务不能再申请和获得任何其他锁。遵守两段封锁协议的事务是可串行化的，但可能发生死锁现象。

封锁粒度：指被封锁的数据对象的大小，有属性、属性集、元组、关系、索引、数据表（库）、物理块等；封锁粒度小则开销大，反之。在进行封锁时综合考虑。

数据库死锁问题：与操作系统的死锁问题类似。

数据库性能优化

对于集中式数据库性能进行优化，可以从以下方面入手：

硬件升级（略）

数据库设计

主要从逻辑设计和物理设计入手。一般数据库规范化程度越高，冗余信息越少，但模式拆解之后使得数据查询的连接越耗时。从某种意义说，非规范化数据库可以改善性能，因此可以适当合理地增加数据冗余。

逻辑设计优化

1、将常用的计算属性（如最大、最小值）存储到数据库实体中。

2、重新定义实体，减少外部属性的数据或行数据的开支。

3、将关系进行水平或垂直分割，提升并行访问程度。

物理设计优化

1、每个属性使用最小存储空间。

2、将一个频繁使用的大关系分割，并存放在两个单独的磁盘中。

3、将文本或图像属性的数据存放在一个单独的物理设备中。

检索策略

建立索引的策略：

1、建立索引应选常用来查询、不经常更新的属性；

2、一个关系上过多的索引会影响UPDATE、INSERT和DELETE性能；

3、找出使用最多的索引进行优化；

4、对数据量非常小的关系不必要建立索引；

查询优化

也成为应用程序优化，是数据库性能优化的最后一个环节，也是最总要一个环节。查询优化策略很多，如：

建立物化视图，减少多表查询
只检索需要的属性
用带IN 的条件字句等价替换OR字句

经常提交（COMMIT），尽早释放锁

等等。

数据库的完整性

数据库完整性是指数据库中数据的正确性和相容性。
可以通过DBMS或应用程序来实现。
基于DBMS的完整性约束作为关系模式的一部分存入数据库中。

完整性约束条件

完整性约束的对象可以是关系、元组或属性三种，以谓词逻辑形式表示，以原子公式和命题连接词（并且、或者、否则）组成逻辑公式。
约束条件值度数据库本身某些语法、语意限制、数据之间逻辑约束以及数据变化应该遵循的规则等。
数据库的完整性约束在关系模式中给出，在运行时做检查。

静态约束

指对数据的语法、语义以及数据之间的逻辑约束，反映数据及其之间的固有逻辑属性，包括：
数据类型的约束
数据格式的约束
取值范围或取值集合的约束
对空值的约束及其他约束

动态约束

反映数据库状态变迁的约束，包括：
修改属性定义的约束
修改属性值时的约束
动态元组约束
动态关系约束

实体完整性

指要求主键中的任一属性不能为空。（更严格的DBMS要求主键不能重复）

参照完整性

一个关系中的外键必须是另一个关系中的主键。
对于完整性约束，删除方式有：
级联删除：指外键值与被参照的关系的主键值相同的元组一起删除，如果参照关系又是另一个关系的被参照关系，这种删除会继续级联下去。
受限删除：一般DBMS的默认删除方式，仅当参照关系中**没有任何元组的外键与**被参照关系中要删除的元组**主键相同**，系统惨允许删除。
置空删除：删除被参照关系的元组，将惨遭关系中元组的**外键置为空值**。

用户定义完整性

指除实体完整性和参照完整性之外，不同的数据库系统根据应用环境不同，提供一些特殊的约束条件。

触发器

触发器指在数据中有违反完整性约束条件时，不仅起到提示作用，还会引起系统自动进行某些操作，以消除违反完整性约束条件所引起的某些负面影响。
触发器中有两种事件:
触发事件：数据的插入、修改、删除操作。
结果事件：一组消除触发事件影响的操作。

数据库安全性

数据库系统安全依赖于：DBMS本身的安全管理措施；应用程序对数据库访问进行的管理和控制。
用户标识和鉴别
系统提供的最外层保护，分为：
口令认证：（略）
强身份认证：结合信息安全领域一些更深入的技术保障措施，如数字证书、智能卡等。

数据授权

对不同的数据角色进行不同级别的授权。一般分三类：
数据库登录权限类：登录、创建视图等。
资源管理权限类：创建表、索引等。
DBA权限类：拥有数据库管理的全部权限。
此外还提供对数据对象的访问控制，分数据库级、关系级、元组级、属性级。

视图

可以看做是虚拟关系或者存储查询。视图在数据库内存储的是SELECT语句，返回构成视图的虚拟关系。视图实现以下功能：
限定元组。

限定属性。
将多个属性连接，是它们看起来像一个关系。
聚合信息而非提供详细信息。

审计与跟踪

审计是一种事后监督手段，使用安全检查措施。它将系统的运行状况和用户访问数据库的行为以日志形式保存，作为一种证据。审计根据对象不同，分：
用户审计：将用户的每一次进行访问的企图（成功或不成功）及每次操作的用户名、时间和操作代码等信息记录下来。

系统审计：记录系统一级的命令和对象的使用情况。

备份与恢复技术

基本原则：保证数据丢失尽量少或完全不丢失，且备份和恢复时间尽量短，保证系统的最大可用性。

物理备份

指在操作系统层面对数据库文件进行备份，分为：
冷备份：也称为静态备份，是指将数据库正常关闭后，对数据库文件备份（复制）。是最快和最安全的方法。
热备份：**利用备份软件**，在数据库正常运行的情况下，将数据文件备份出来。
通常结合完全、增量和差异三种备份方式，已一个备份周期和多个增量、差异备份组成。

逻辑备份

利用DBMS自带的工具备份和恢复数据库内容，可以按照表、表空间、用户和全库等4个层次备份和恢复数据。在数据库容量并不大的情况下，是一种非常有效的手段。

日志文件

事务日志记录了数据库的任何操作，可以根据这些记录将数据文件恢复成事务前的状态。
热备份中必须建立日志文件，同时遵循"先写日志再写文件"的规则。

数据恢复

将数据库从错误状态恢复到某一个已知的正确状态的功能，称为数据库的恢复。根据不同的故障类型，采取不同的恢复策略：
事务故障和恢复：反向扫描事务日志，找到更新操作并执行逆操纵，直至事务开始。（系统自动完成）
系统故障和恢复：
介质故障与病毒破坏的恢复：
有检查点的恢复：

数据中心

数据库设计与建模

基于数据库系统生命周期的数据库设计可以分为5个阶段：

规划

主要是建立数据库必要性、可行性分析，确定数据库系统在信息系统中的地位，以及各个数据库之间的 联系。

需求分析

通过调查研究，了解数据和处理要求，形成需求说明书。
通过对数据名、数据类型、保密要求，完整性约束等数据的管理，形成数据字典。

概念设计

也称概念结构设计，根据需求分析产生的需求说明书的基础上，将它们抽象为一个不依赖DBMS的数据模型，称为数据模型。

逻辑设计

目的是将概念模型转化为某个特定的DBMS的逻辑模型，如将E-R图转换为选用的具体机器上的DBMS所支持的数据模型相符合的逻辑结构。

物理设计

任务是对给定的逻辑模型选其一个适合的应用环境的物理结构，主要是存储结构和存取方法。

实体联系模型（E-R模型）

见知识点汇总--数据库--实体联系模型。

数据库规范化

第一范式 | 第二范式 | 第三范式
BC范式 | 第四范式 | 第五范式 | DK范式 | 第六范式
反规范化

==========

参考资料：

数据库管理系统（DBMS）（查看 • 讨论 • 编辑 • 历史）

概念

数据库 · 数据模型 · 数据库存储结构 · 关系（数据库） · 关系模型 · 分布式数据库 · ACID（原子性 · 一致性 · 隔离性 · 持久性） · Null值

关系数据库 · 关系代数 · 关系演算 · 元组关系演算 · 域关系演算 · 数据库规范化 · 引用完整性 · 关系数据库管理系统

主键 · 外键 · 代理键 · 超键 · 候选键

数据库组件	*SQL*
触发器 · 视图 · 数据库表 · 指标（数据库） · 事务日志 · 数据库事务 · 并发控制 · 乐观锁 · 悲观锁 · 数据库索引 存储程序 · 数据库分区	分类： 数据查询语言（DQL） - 数据定义语言（DDL） - 数据操纵语言（DML） - 数据控制语言（DCL） 指令：SELECT · INSERT · UPDATE · MERGE · DELETE · JOIN · UNION · CREATE · DROP · Begin work · COMMIT · ROLLBACK · TRUNCATE · ALTER 安全： SQL注入攻击 · 参数化查询

数据库管理系统的实施

实施类型

关系数据库 · 文件型数据库 · Deductive · 维度化数据库 · 层次结构式 · 图形数据库 · 对象数据库 · 对象关系数据库 · Temporal · XML数据库

数据库产品	数据库成分
对象型（对比） · 关系型（对比）	数据查询语言 · 查询优化器 · 查询计划 · 嵌入式SQL · ODBC · JDBC · OLE DB