复习提纲

第 01 章 绪论

1、数据（Data）：描述事物的符号记录，是数据库中存储的基本对象。

数据的含义称为数据的语义，数据与其语义是不可分的。

2、数据库（Database，简称 DB）是长期储存在计算机内、有组织的、可共享的大量数据的集合。

基本特征：1）数据按一定的数据模型组织、描述和储存

2）可为各种用户共享

3）冗余度较小

4）数据独立性较高（和应用程序无关）

5）易扩展

3、数据库管理系统-DBMS 是位于用户与操作系统之间的一层数据管理软件

4、数据库系统（Database System，简称 DBS）：由数据库、数据库管理系统、应用程序和数据库管理员组成的存储、管理、处理和维护数据的系统。

1）数据库提供数据的存储功能；

2）数据库管理系统提供数据的组织、存取、管理和维护等功能；

3）数据库应用系统根据应用需求使用数据库；

4）数据库管理员负责全面管理数据库系统

5、数据库管理技术经历了人工管理、文件系统、数据库系统三个阶段。

6、数据库系统的特点：选择题

1）数据结构化；

2）数据的共享性搞、冗余度低、易扩展；

3）数据独立性高；

4）数据由数据库管理系统统一管理和控制。

7、数据模型是对现实世界数据特征的抽象，是数据库系统的核心和基础。

8、概念模型也称信息模型，它是按用户的观点来对数据和信息建模，用于数据库设计。

9、逻辑模型和物理模型

逻辑模型主要包括网状模型、层次模型、关系模型、面向对象数据模型、对象关系数据模型、半结构化数据模型等。按计算机系统的观点对数据建模，用于 DBMS 实现。

物理模型是对数据最底层的抽象，描述数据在系统内部的表示方式和存取方法，在磁盘或磁带上的存储方式和存取方法。 填空题

10、概念模型中的基本概念：

1）实体：客观存在并可相互区别的事物称为实体。

2）属性：实体所具有的某一特性称为属性。

3）唯一标识实体的属性集称为码。

4）联系：现实世界中事物内部以及事物之间的联系在信息世界中反映为实体（型）内部的联系和实体（型）之间的联系。

实体之间的联系有一对一、一对多和多对多等多种类型。

11、概念模型的表示方法：E-R 模型

12、数据模型的组成要素：数据结构、数据操作、数据的完整性约束条件

13、常用的数据模型：

1）层次模型：数据库系统中最早出现的数据模型。

用树形结构来表示各类实体以及实体间的联系。

只能处理一对多的实体联系。

典型代表：IBM 公司的 IMS

2）网状模型：用网状结构来表示各类实体以及实体间的联系。

能处理多对多的实体联系。层次模型是网状模型的特例。

典型代表：DBTG 系统

3）关系模型：是最重要的一种数据模型。

在用户观点下，关系模型中数据的逻辑结构是一张二维表，它由行和列组成。

关系：一个关系对应通常说的一张表。

元组：表中的一行即为一个元组。

属性：表中的一列即为一个属性，给每一个属性起一个名称即属性名

主码：也称码键。表中的某个属性组，它可以唯一确定一个元组

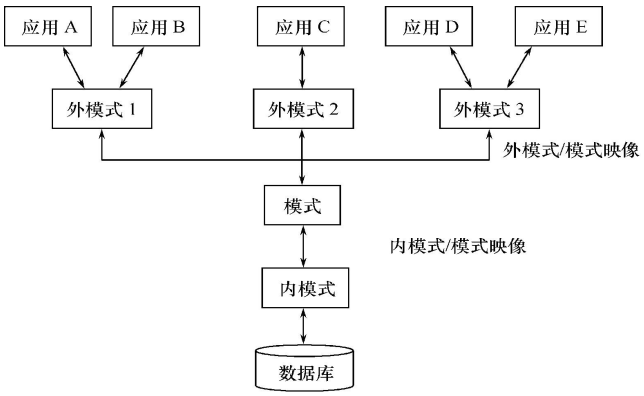
域：是一组具有相同数据类型的值的集合。属性的取值范围来自某个域。

分量：元组中的一个属性值

关系模式：关系名（属性 1，属性 2，…，属性 n）

完整性约束条件：实体完整性、参照完整性、用户定义的完整性

14、三级模式：外模式、模式、内模式



1）模式：也称逻辑模式，是数据库中全体数据的逻辑结构和特征的描述，是所有用户的公共数据视图。

一个数据库只有一个模式。

2）外模式：也称子模式/用户模式，是数据库用户能够看见和使用的局部数据的逻辑结构和特征的描述，是数据库用户的数据视图。

一个数据库可以有多个外模式

3）内模式：也称存储模式，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组织方式。

一个数据库只有一个内模式

15、二级映像：外模式/模式映像、模式/内模式映像

1）外模式/模式映像：保证了数据与程序的逻辑独立性

2）模式/内模式映像：保证了数据与程序的物理独立性

16、数据库系统的组成：

数据库
数据库管理系统
应用程序
数据库管理员

第 02 章 关系数据库

1、关系：现实世界的实体以及实体间的各种联系均用关系来表示
   1）域：是一组具有相同数据类型的值的集合
   2）笛卡尔积：域上的一种集合运算
      $D_1 \times D_2 \times \cdots \times D_n = \{(d_1, d_2, \cdots, d_n) \mid d_i \in D_i, i = 1, 2, \cdots, n\}$
   3）元组：笛卡尔积中每一个元素（d1，d2，…，dn）叫作元组
   4）基数：一个域允许的不同取值个数
   5）关系：$D_1 \times D_2 \times \cdots \times D_n$的子集叫作在域 D_1，D_2，…，D_n上的关系，表示为
      $R(D_1, D_2, \cdots, D_n)$
2、关系的三种类型：
   1）基本表
   2）查询表
   3）视图表
3、 基本关系的 6 条性质
   1） 列是同质的（来自同一个域）
   2） 不同的列可以出自同一个域，不同的属性要给予不同的属性名
   3） 列的次序可以任意交换
   4） 行的次序可以任意交换
   5） 任意两个元组的候选码不能取相同的值
   6） 分量必须是原子值
4、关系模式：形式化地表示为：R（U，D，DOM，F） 简写为 R(U)

R	关系名
U	组成该关系的属性名集合
D	U 中属性所来自的域
DOM	属性向域的映象集合
F	属性间数据的依赖关系的集合

5、关系操作：查询操作和更新操作（插入、删除、修改）两大部分
6、查询的五种基本操作：填空题
   选择、投影、并、差、笛卡尔积
7、关系数据语言
   1）关系代数语言
8、关系的完整性：是对关系的某种约束条件
   实体完整性：主属性唯一，且不能为空值
   参照完整性：定义外码和主码之间的引用关系，外码可以取空值或者某个主码值
   用户定义的完整性：反映某一具体应用所涉及的数据必须满足的语义要求
9、关系代数运算

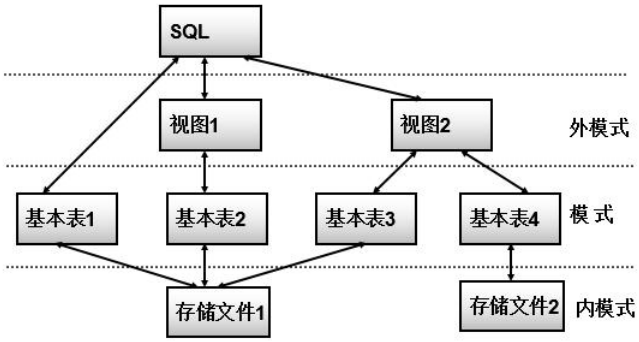
运 算 符	含 义

集合运算符	∪	并
	-	差
	∩	交
	×	笛卡尔积
专门的关系运算符	σ	选择
	π	投影
		连接
	÷	除

例：2.4、2.5、2.6、2.11、2.12、2.13
习题：6

第 03 章关系数据库标准语言 SQL

1、SQL 结构化查询语言，是关系数据库的标准语言
2、SQL 集数据查询、数据操纵、数据定义、数据控制功能于一体。其特点：
　　1）综合统一（实体和实体间的联系都用关系表示，每一种操作只需一种操作符）
　　2）高度非过程化（有利于提高数据独立性）
　　3）面向集合的操作方式（元组的集合）
　　4）同一种语法结构提高多种使用方式（独立语言，也可以嵌入式语言）
　　5）语言简洁，易学易用
3、SQL 支持关系数据库三级模式结构



4、学生-课程模式
　　学生表：Student(Sno,Sname,Ssex,Sage,Sdept)
　　课程表：Course(Cno,Cname,Cpno,Ccredit)
　　学生选课表：SC(Sno,Cno,Grade)
5、现代关系数据库管理系统提供了一个层次化的数据库对象命名机制
　　一个关系数据库管理系统的实例（Instance）中可以建立多个数据库
　　一个数据库中可以建立多个模式
　　一个模式下通常包括多个表、视图和索引等数据库对象
6、模式的定义与删除
　　1）CREATE SCHEMA “S-T” AUTHORIZATION WANG;
　　2）DROP SCHEMA <模式名> <CASCADE|RESTRICT>
　　　　CASCADE（级联）：删除模式的同时把该模式中所有的数据库对象全部删除

RESTRICT（限制）：

- 如果该模式中定义了下属的数据库对象（如表、视图等），则拒绝该删除语句的执行。
- 仅当该模式中没有任何下属的对象时才能执行。

7、基本表的定义、删除与修改

1）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>]);

例如：3.5、3.6、3.7

2）数据类型：表明属性的类型和长度

char(n)、varchar(n)、int、smallint、numeric(p,d)、boolean 等

3）ALTER TABLE <表名>

[ADD[COLUMN] <新列名> <数据类型> [完整性约束]]

[ADD <表级完整性约束>]

[DROP [COLUMN] <列名> [CASCADE| RESTRICT]]

[DROP CONSTRAINT<完整性约束名>[RESTRICT | CASCADE]]

[ALTER COLUMN <列名><数据类型>] ;

例如：3.8、3.9、3.10

4）DROP TABLE <表名>［RESTRICT| CASCADE］

8、索引的建立与删除

1）建立索引的目的：加快查询速度

2）常见索引：顺序文件的索引、B+树索引、散列索引、位图索引

3）数据库管理员 或 表的属主（即建立表的人）建立索引，关系数据库管理系统自动完成索引

4）CREATE [UNIQUE] [CLUSTER] INDEX <索引名>

ON <表名>(<列名>[<次序>][,<列名>[<次序>]]…)

5）ALTER INDEX <旧索引名> RENAME TO <新索引名>

6）DROP INDEX <索引名>

9、单表查询

1）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>]…|(SELECT 语句)

[AS]<别名>

[WHERE <条件表达式>]

[GROUP BY <列名 1> [HAVING <条件表达式>]]

[ORDER BY <列名 2> [ASC|DESC]];

例如：3.16、3.18、3.19、3.21、3.25、3.30、3.38、3.40、3.41、3.43、3.46、3.48

2）查询条件

查 询 条 件	谓　词
比　较	=, >, <, >=, <=, !=, <>, !>, !<; NOT+上述比较运算符
确定范围	BETWEEN AND, NOT BETWEEN AND
确定集合	IN, NOT IN
字符匹配	LIKE, NOT LIKE
空　值	IS NULL, IS NOT NULL
多重条件（逻辑运算）	AND, OR, NOT

3）聚集函数

count()、sum()、avg()、max()、min()

10、连接查询：同时涉及两个以上的表的查询

1）例如 3.51、3.53

11、嵌套查询：将一个查询块嵌套在另一个查询块的 WHERE 子句或 HAVING 短语的条件中的查询

1）上层的查询块称为外层查询或父查询，下层查询块称为内层查询或子查询

2）子查询中不能使用 ORDER BY 子句

3）例：3.57、3.59

4）

使用ANY或ALL谓词时必须同时使用比较运算

语义为：

> ANY	大于子查询结果中的某个值
> ALL	大于子查询结果中的所有值
< ANY	小于子查询结果中的某个值
< ALL	小于子查询结果中的所有值
>= ANY	大于等于子查询结果中的某个值
>= ALL	大于等于子查询结果中的所有值

例：3.60

5）带有 EXISTS 谓词的子查询不返回任何数据，只产生逻辑真值“true”或逻辑假值“false”。

例：3.62

6）集合操作的种类

并操作 UNION

交操作 INTERSECT

差操作 EXCEPT

例：3.66、3.68

7）子查询不仅可以出现在 WHERE 子句中，还可以出现在 FROM 子句中，这时子查询生成的临时派生表（Derived Table）成为主查询的查询对象

12、数据插入

1）INSERT　INTO <表名> [(<属性列 1>[,<属性列 2 >…)]

VALUES (<常量 1> [,<常量 2>]…);

例如：3.71

2）插入子查询结果

INSERT　INTO <表名>　[(<属性列 1> [,<属性列 2>…　)]

子查询;

例如：3.74

13、数据修改

1）UPDATE <表名>

SET <列名>=<表达式>[,<列名>=<表达式>]…

[WHERE <条件>];

例如：3.75

14、数据删除

1）DELETE FROM <表名>

[WHERE <条件>];

例如：3.78

15、空值处理

1）空值就是“不知道”或“不存在”或“无意义”的值

2）判断一个属性的值是否为空值，用 IS NULL 或 IS NOT NULL 来表示

3）属性定义中有 NOT NULL 约束条件的不能取空值，码属性不能取空值

16、视图：虚表，从一个或几个基本表（或视图）导出的表，只存放视图的定义，不存放视图对应的数据。

1）CREATE VIEW <视图名> [(<列名> [,<列名>]…)]

AS <子查询>

[WITH CHECK OPTION];

例如：3.86

2）DROP VIEW <视图名>[CASCADE];

例如：3.93

3）查询视图与查询基本表相同

例如：3.94

4）更新视图：通过视图来 insert delete update 数据，实际是对基本表的更新

例如：3.97

并不是所有视图都是可更新的。

5）视图的作用：

- 视图能够简化用户的操作
- 视图使用户能以多种角度看待同一数据
- 视图对重构数据库提供了一定程度的逻辑独立性
- 视图能够对机密数据提供安全保护
- 适当的利用视图可以更清晰的表达查询

习题：5（1-8）

第 04 章数据库安全性

1、数据库的数据保护主要包括数据的安全性和数据的完整性。

2、数据库的安全性是指保护数据库以防止不合法使用所造成的数据泄露、更改或破坏。

3、数据库的不安全因素

1）非授权用户对数据库的恶意存取和破坏

2）数据库中重要或敏感的数据被泄露

3）安全环境的脆弱性

4、安全标准 TCSEC/TDI 的四个指标：安全策略、责任、保证、文档

5、TCSEC/TDI 安全级别划分

安全级别	定　义
A1	验证设计（Verified Design）
B3	安全域（Security Domains）
B2	结构化保护（Structural Protection）
B1	标记安全保护（Labeled Security Protection）
C2	受控的存取保护（Controlled Access Protection）
C1	自主安全保护（Discretionary Security Protection）
D	最小保护（Minimal Protection）

6、数据库安全性控制

1）安全技术：用户身份鉴别、多层存取控制、审计、视图、数据加密

2）用户身份鉴别方法：

静态口令鉴别

动态口令鉴别

生物特征鉴别

智能卡鉴别

3）存取控制机制包括用户权限和合法权限检查两部分，

4）自主存取控制：通过 SQL 的 GRANT 语句和 REVOKE 语句实现

GRANT 语句的一般格式：

GRANT <权限>[,<权限>]...

ON <对象名>[, <对象名>]…

TO <用户>[,<用户>]...

[WITH GRANT OPTION];

例如：4.1、4.2、4.4

按受权限的用户：可以是一个或多个用户，也可以是 Public(全体用户)，

不允许循环授权

5）REVOKE <权限>[,<权限>]...

ON　<对象名>[,<对象名>]…

FROM <用户>[,<用户>]...[CASCADE | RESTRICT];

例如：4.8、4.9

6）创建用户：CREATE　USER　<username> [WITH][DBA|RESOURCE|CONNECT]

CONNECT 权限的用户不能创建新用户，不能创建模式，也不能创建基本表，只能登录数据库

RESOURCE 权限的用户能创建基本表和视图，不能创建模式，不能创建新的用户

DBA 权限的用户是系统中的超级用户

7）数据库角色：被命名的一组与数据库操作相关的权限，角色是权限的集合。

可以为一组具有相同权限的用户创建一个角色。

CREATE　ROLE　<角色名>

GRANT　<权限>[,<权限>]…

ON　对象名

TO <角色>[,<角色>]…

习题：5、6

第 05 章数据库完整性

1、数据库的完整性是指数据的正确性和相容性。

2、实体完整性在 CREATE　TABLE 中用 PRIMARY KEY 定义。

1）单属性构成的码可以定义为列级约束条件和表级约束条件。

2）多属性构成的码只能定义为表级约束条件

3）实体完整性规则自动进行检查：

检查主码值是否唯一；

检查主码的各个属性是否为空

例如：5.1

3、参照完整性定义在 CREATE　TABLE 中用 FOREIGN KEY 短语定义哪些列为外码，用 REFERENCES 短语指明这些外码参照哪些表的主码。

1）对被参照表和参照表进行增删改操作时有可能破坏参照完整性，必须进行检查

表5.1 可能破坏参照完整性的情况及违约处理

被参照表（例如Student）	参照表（例如SC）	违约处理
可能破坏参照完整性 ←	插入元组	拒绝
可能破坏参照完整性 ←	修改外码值	拒绝
删除元组 →	可能破坏参照完整性	拒绝/级连删除/设置为空值
修改主码值 →	可能破坏参照完整性	拒绝/级连修改/设置为空值

例如：5.3

4、用户定义的完整性：是针对某一具体应用的数据必须满足的语义要求

1）CREATE TABLE 时定义属性上的约束条件

- 列值非空（NOT NULL）
- 列值唯一（UNIQUE）
- 检查列值是否满足一个条件表达式（CHECK）

例如：5.5、5.6、5.7、5.8

2）在 CREATE TABLE 时可以用 CHECK 短语定义元组上的约束条件，即元组级的限制

例如：5.9

5、完整性约束命名子句

1） CONSTRAINT <完整性约束条件名><完整性约束条件>

<完整性约束条件>包括 NOT NULL、UNIQUE、PRIMARY KEY 短语、FOREIGN KEY 短语、CHECK 短语等

例如 5.10

2） 使用 ALTER TABLE 语句修改表中的完整性限制

例如：5.13

6、触发器

1）触发器（Trigger）是用户定义在关系表上的一类由事件驱动的特殊过程

2）触发器保存在数据库服务器中，用户对表的增、删、改操作均由服务器自动激活相应的触发器

3）create trigger 触发器名　　(SQL SERVER 中的写法)

on 表名　instead of/after　触发事件（insert、update、delete）

As

触发动作体

例如：5.18(改为 SQL SERVER 中的写法）

4）触发器只能定义在基本表上，不能定义在视图上

5）触发器的执行，是由触发事件激活的，并由数据库服务器自动执行，一个数据表上可以定义多个触发器

6）DROP TRIGGER <触发器名>

习题：6、8（使用触发器）

第 06 章关系数据理论

1、数据库的规范化理论：数据库逻辑设计的有力工具

2、关系模式看作一个三元组：R<U,F>，U 为一组属性，F 为属性组 U 上的一组数据依赖

3、数据依赖：是一个关系内部属性与属性之间的一种约束关系

1）函数依赖

2）多值依赖

4、一个“好”的模式应当不会发生插入异常、删除异常和更新异常，数据冗余应尽可能少。

5、函数依赖

1）定义 6.1　设 R(U)是一个属性集 U 上的关系模式，X 和 Y 是 U 的子集。若对于 R(U)的任意一个可能的关系 r，r 中不可能存在两个元组在 X 上的属性值相等， 而在 Y 上的属性值不等， 则称“X 函数确定 Y”或“Y 函数依赖于 X”，记作 X→Y。

2）X→Y，但 Y⊈X 则称 X→Y 是非平凡的函数依赖。

3）X→Y，但 Y⊆X 则称 X→Y 是平凡的函数依赖。

4）定义 6.2　在 R(U)中，如果 X→Y，并且对于 X 的任何一个真子集 X’，都有 X’ ↛ Y，则称 Y 对 X 完全函数依赖，记作 $X \xrightarrow{F} Y$。

5）若 X→Y，但 Y 不完全函数依赖于 X，则称 Y 对 X 部分函数依赖，记作 $X \xrightarrow{P} Y$

6）定义 6.3　在 R(U)中，如果 X→Y(Y⊈X)，Y↛X，Y→Z，Z⊈Y，则称 Z 对 X 传递函数依赖(transitive functional dependency)。记为：$X \xrightarrow{传递} Z$。

6、码

1）定义 6.4　设 K 为 $R<U,F>$ 中的属性或属性组合。若 $K \xrightarrow{F} U$，则 K 称为 R 的一个候选码(Candidate Key)。

2）如果 U 部分函数依赖于 K，即 $K \xrightarrow{P} U$,则 K 称为超码（Surpkey）。候选码是最小的超码，即 K 的任意一个真子集都不是候选码。

3）若关系模式 R 有多个候选码，则选定其中的一个做为主码(Primary key)。

4）定义 6.5　关系模式 R 中属性或属性组 X 并非 R 的码，但 X 是另一个关系模式的码，则称 X 是 R 的外部码（Foreign key）也称外码。

7、范式是符合某一种级别的关系模式的集合

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

8、1NF：每一个分量必须是不可分的数据项

9、2NF：

1）定义 6.6　若关系模式 R∈1NF，并且每一个非主属性都完全函数依赖于任何一个候

选码，则 $R \in 2NF$

例如：6.4

10、3NF

1）定义 6.7　设关系模式 R<U,F>∈1NF,若 R 中不存在这样的码 X、属性组 Y 及非主属性 Z（Z ⊇ Y），使得 X→Y，Y→Z 成立，Y ↛ X 不成立，则称 R<U,F> ∈ 3NF。

11、BCNF

1）定义 6.8　设关系模式 R<U,F>∈1NF，若 X →Y 且 Y ⊆ X 时 X 必含有码，则 R<U,F>∈BCNF。

- 所有非主属性都完全函数依赖于每个候选码
- 所有主属性都完全函数依赖于每个不包含它的候选码
- 没有任何属性完全函数依赖于非码的任何一组属性

习题：2、6

第 07 章数据库设计

1、数据库设计：指对于一个给定的应用环境，构造（设计）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，包括信息管理要求和数据操作要求。

2、数据库设计的目标是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 。

3、数据库设计特点：1）三分技术，七分管理，十二分基础数据 ；2）结构设计和行为设计相结合

4、数据库设计要求多方面的知识和技术，常见的设计方法：新奥尔良、E-R 模型、3NF、UML 等

5、数据库设计分 6 个阶段

需求分析（基础）、概念结构设计（概念模型-关键）、逻辑结构设计（数据模型）、物理结构设计（物理结构）、数据库实施、数据库运行和维护

6、需求分析

1）分析用户的要求：信息要求、处理要求、安全性与完整性要求

2）与用户不断深入的交流，逐步确定用户的实际需求

3）分析方法：结构化分析方法（SA）采用自顶向下、逐层分解的方式

7、数据字典

1）是关于数据库中数据的描述，即元数据，不是数据本身

2）在需求分析阶段建立

3）包括：数据项、数据结构、数据流、数据存储和处理过程

8、概念模型

1）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

2）描述概念结构的工具：E-R 模型（实体、属性、联系）

3）实体间的联系：

一对一 1：1、一对多 1：n、多对多 m：n

4）E-R 图：

实体用矩形表示、属性用椭圆表示、联系用菱形表示

5）例题：学生学籍管理子系统的 E-R 图

6）实体与属性划分原则：

作为属性，不能再具有需要描述的性质，必须是不可分的数据项，不能包含其他属性

属性不能与其它实体具有联系

9、逻辑结构设计

1）将 E-R 图转换为关系模型：将实体型、实体的属性和实体型之间的联系转化为关系模式

2）转换原则：

一个实体型转换为一个关系模式，关系的属性就是实体的属性，关系的码就是实体的码

3）联系的转换：

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

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

一个 *m*:*n* 联系转换为一个关系模式

4）数据库逻辑设计的结果不是唯一的，关系数据模型的优化通常以规范化理论为指导

5）对关系模式进行必要分解，提高数据操作效率和存储空间的利用率。分为水平分解和垂直分解。

6）定义用户子模式（外模式），主要考虑用户的习惯和方便：

使用更符合用户习惯的别名；

对不同级别的用户定义不同的视图，以保证系统的安全性；

简化用户对系统的使用。

10、物理结构设计

1）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

2）对物理结构评价的重点：时间和空间效率

3）数据库运行阶段，数据库的维护工作由数据库管理员完成：

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、分析和改造

数据库的重组织与重构造

习题：7、8

第 08 章数据库编程

1、SQL 语言提供了 2 种使用方式：交互式和嵌入式

2、嵌入式 SQL：将 SQL 语句嵌入程序设计语言中。比如 C、C++等（宿主语言）

1）采用预编译方法处理嵌入式 SQL

2）与主语言之间的通信：

SQL 通信区、主变量、游标

2、过程化 SQL：SQL 的扩展，增加了过程化语句功能

1）基本结构是块，所有的过程化 SQL 程序都由块组成，每个块完成一个逻辑操作。

2）变量和常量的定义、赋值语句、条件控制语句、循环控制语句、错误处理

3）过程化 SQL 块有两种类型：命名块和匿名块

3、存储过程

1）由过程化 SQL 语句书写的过程，经编译和优化后存储在数据库服务器中，使用时只要调用即可。

2）优点：（1）运行效率高、（2）降低了客户机和服务器之间的通信量、（3）方便实施企业规则

3）创建存储过程

CREATE PROCEDURE 过程名([参数 1,参数 2,...])

AS

<过程化 SQL 块>

例题：8.5（修改为 SQL SERVER 语句）

4）执行存储过程

exec 存储过程名（参数 1……）

3） 删除存储过程

drop 存储过程名

4、函数

1）函数和存储过程类似，不同之处是函数必须指定返回的类型

2）创建函数

CREATE FUNCTION 函数名 ([参数 1,参数 2,…]) RETURNS <类型>

Begin

<过程化 SQL 块>

End

例题：自定义函数：查询选修某门课的学生的平均成绩

习题：2(①③用存储过程，②用函数)

第 09 章关系数据库存储管理

1、数据组织

1）逻辑组织方式：表空间-段-分区-数据块

一个表空间对应磁盘上一个或多个物理文件，但一个物理文件只能属于一个表空间；

2）物理组织方式：文件-块-记录

3）记录表示：定长记录存储：每条记录占据相同大小的空间

变长记录存储：每条记录占据大小不相同的空间

4）关系表的组织：堆存储

顺序存储

多表聚簇存储

B+树存储

哈希存储

5）索引结构：顺序表索引、辅助索引、B+树索引、哈希索引、位图索引

第 10 章关系查询处理和查询优化

1、查询处理

1）查询处理：是关系数据库管理系统执行查询语句的过程；任务是把用户提交的查询

语句转换为高效的查询执行计划。

2）分为 4 个阶段：

查询分析、查询检查、查询优化 、查询执行

3）查询分析的任务：对查询语句进行扫描、词法分析和语法分析

4）查询检查的任务：做语义分析、视图转换等

5）查询优化：代数优化和物理优化

6）查询执行：依据优化器得到的执行策略生成查询执行计划，由代码生成器生成代码执行

7）选择操作中只涉及一个关系，一般采用全表扫描或者基于索引的算法实现查询。

2、查询优化

1）关系查询优化是影响关系数据库管理系统性能的关键因素

2）查询优化的优点：用户不必考虑如何最好地表达查询以获得较好的效率，由系统决定

3）选择和连接操作时，先做选择操作，这样参加连接的元组就可以大大减少

3、代数优化

1）通过对关系代数表达式的等价变换来提高查询效率

2）常用的等价变换规则：

(1) 连接、笛卡尔积交换律
(2) 连接、笛卡尔积的结合律
(3) 投影的串接定律
(4) 选择的串接定律
(5) 选择与投影操作的交换律
(6) 选择与笛卡尔积的交换律
(7) 选择与并的分配律
(8) 选择与差运算的分配律
(9) 选择对自然连接的分配律
(10) 投影与笛卡尔积的分配律
(11) 投影与并的分配律

习题：4

第 11 章数据库恢复技术

1、事务

1）事务(Transaction)是用户定义的一个数据库操作序列，这些操作要么全做，要么全不做，是一个不可分割的工作单位。

2）一个程序通常包含多个事务

3）定义事务的语句：begin transaction、commit、rollback

4）事务的 ACID 特性：

原子性
一致性
隔离性
持续性

2、数据库恢复：数据库管理系统必须具有把数据库从错误状态恢复到某一已知的正确状态

(亦称为一致状态或完整状态)的功能

3、故障的种类

1）事务内部的故障：事务程序本身发现的故障和非预期，不能由应用程序处理的故障。

采用的恢复策略：事务撤销（UNDO）

2）系统故障：称为软故障，是指造成系统停止运转的任何事件，使得系统要重新启动。

采用的恢复策略：事务撤销（UNDO）和重做（REDO）

3）介质故障：称为硬故障，指外存故障（磁盘损坏等）

4）计算机病毒：一种人为的故障或破坏，是一些恶作剧者研制的一种计算机程序

4、恢复的实现技术

1）建立冗余数据最常用的技术：数据转储和登记日志文件

2）转储是指数据库管理员定期地将整个数据库复制到磁带、磁盘或其他存储介质上保存起来的过程

3）转储分为：静态存储和动态存储

4）需要把动态转储期间各事务对数据库的修改活动登记下来，建立日志文件

5）转储也分为：海量存储和增量存储

6）日志文件(log file)是用来记录事务对数据库的更新操作的文件

两种格式：以记录为单位的日志文件

以数据块为单位的日志文件

7）登记日志文件遵循的原则：

（1）登记的次序严格按并发事务执行的时间次序；

（2）先写日志文件，后写数据库

5、恢复策略

1）事务故障的恢复：由系统自动完成，对用户是透明的，不需要用户干预

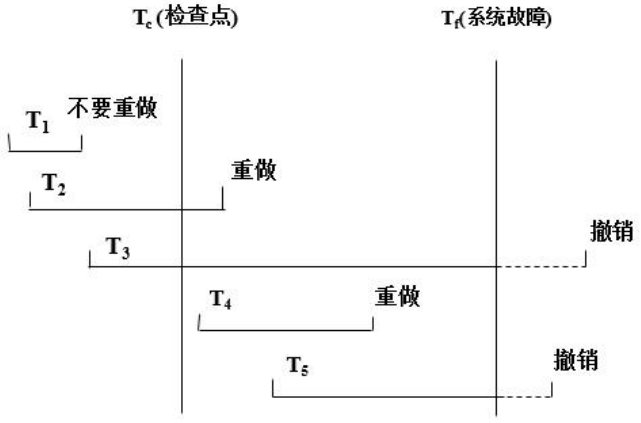
2）系统故障的恢复：由系统在重新启动时自动完成，不需要用户干预

3）介质故障的恢复：重装数据库，重做已完成的事务，需要数据库管理员介入

6、具有检查点的恢复技术

1）在日志文件中增加检查点记录，增加重新开始文件，并让恢复子系统在登录日志文件期间动态地维护日志

2）系统出现故障时，恢复子系统将根据事务的不同状态采取不同的恢复策略



习题：4, 5

第 12 章并发控制

1、并发控制

1）为了充分利用系统的资源，发挥数据库共享资源的特点，应该允许多个事务并行地执行。

2）在单处理机系统中，事务的并行执行是这些并行事务的并行操作轮流交叉运行

3）事务是并发控制的基本单位

4）并发操作带来的数据不一致性包括丢失修改、不可重复读和读“脏”数据

T_1	T_2
① R(A)=16	
②	R(A)=16
③ A←A-1	
W(A)=15	
④	A←A-1
	W(A)=15

丢失修改

T_1	T_2
① R(A)=50	
R(B)=100	
求和=150	
②	R(B)=100
	B←B*2
	W(B)=200
③ R(A)=50	
R(B)=200	
求和=250	
(验算不对)	

不可重复读

T_1	T_2
① R(C)=100	
C←C*2	
W(C)=200	
②	R(C)=200
③ ROLLBACK	
C恢复为100	

读“脏”数据

5）并发控制的主要技术：封锁、时间戳、乐观控制法、多版本并发控制等

2、封锁

1）事务 T 在对某个数据对象（例如表、记录等）操作之前，先向系统发出请求，对其加锁，称为封锁

2）封锁的类型：排他锁（X 锁/写锁）和共享锁（S 锁/读锁）

3、封锁协议

1）在运用 X 锁和 S 锁对数据对象加锁时，需要约定一些规则，这些规则为封锁协议

2）一级封锁协议：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。

一级封锁协议可防止丢失修改，并保证事务 T 是可恢复的。

3）二级封锁协议：一级封锁协议加上事务 T 在读取数据 R 之前必须先对其加 S 锁，读完后即可释放 S 锁。

二级封锁协议可以防止丢失修改和读“脏”数据

4）三级封锁协议：一级封锁协议加上事务 T 在读取数据 R 之前必须先对其加 S 锁，直到事务结束才释放。

三级封锁协议可防止丢失修改、读脏数据和不可重复读。

4、活锁和死锁

1）事务 T 处于永远等待状态，称为活锁

解决策略：先来先服务

2）事务 T1 和 T2 处于互相等待状态，形成死锁