需要记忆的东西

* （不重要）DBMS定义，数据库定义，数据库系统定义，DBA职责
* 三级模式的数据独立性
* 四种数据模型对比
* 空值null
* 视图特点，与表的区别，作用
* 完整性约束，参照完整性

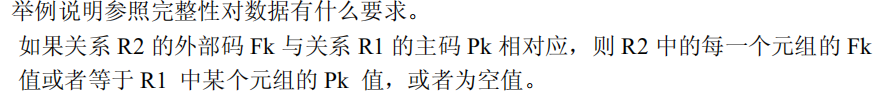
完整性约束：

数据库完整性（Database Integrity）是指数据库中数据的正确性、有效性和相容性。数据库完整性由各种各样的完整性约束来保证，因此可以说数据库完整性设计就是数据库完整性约束的设计。

1、 实体完整性：要求每个关系模式有且仅有一个主码，每个主码的值必须唯一，而且不能为空。

2、 域完整性：数据库表中的列必须满足某种特定的数据类型或约束。其中约束又包括取值范围、精度等规定。表中的CHECK、FOREIGN KEY 约束和DEFAULT、 NOT NULL定义都属于域完整性的范畴。

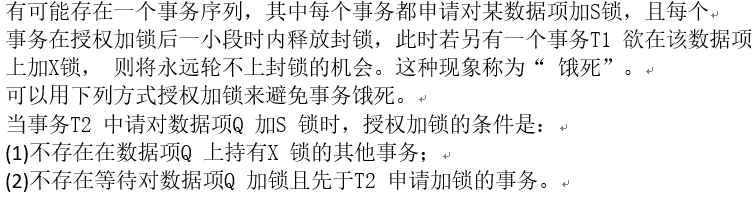
3、 参照完整性：参照的完整性要求关系中不允许引用不存在的实体。当更新、删除、插入一个表中的数据时，通过参照引用相互关联的另一个表中的数据，来检查对表的数据操作是否正确。

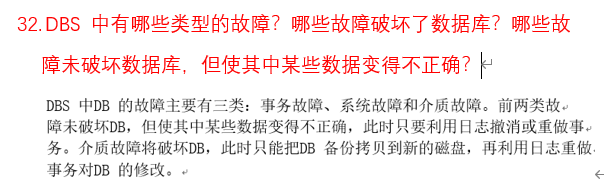


* 写sql，关系代数，元组演算（书上笔记易错点）：

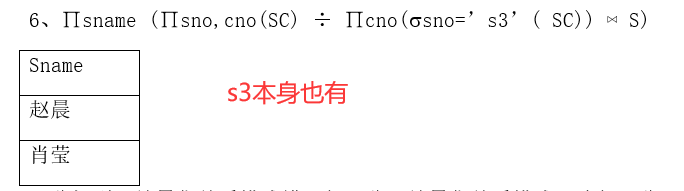
三种表示全部的方法；update，delete，grant（授权），create（表，视图，索引），drop，alter，join写法，数据类型（大数据类型），交并补

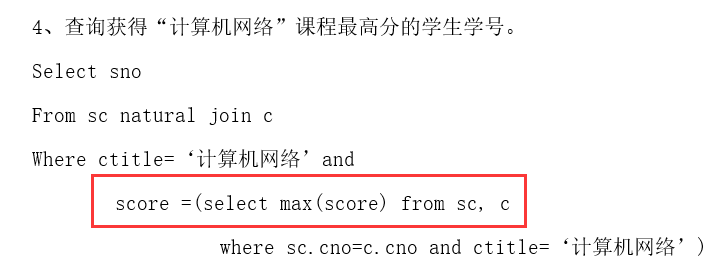
* 写jdbc，背
* 弱实体集定义：如果一个实体集的所有属性都不足以形成主码，则称这样的实体集为弱
* 实体集。每个弱实体集必须与另一个称作标识或主实体集的实体集关联才能有意义。
* 聚集定义
* 函数依赖和多值依赖区别
* 正则覆盖定义，无关属性定义
* BCNF定义（验证是否满足BCNF，只用F就够了），3NF定义，4NF定义
* 需要记住过程的：求闭包的armstrong公理；求正则覆盖算法；求属性集闭包算法，保持依赖验证算法；BCNF分解算法（特点分析）；3NF分解算法
* 各种索引：主&辅助，稠密&稀疏，顺序&散列，多级
* B+树索引特点
* 查询优化代价计算（书上的表）
* 查询表达tree优化，等价表达式，书上（主要就是俩原则）
* 事务的概念以及ACID特性
* 可串行化和可恢复的调度各是什么
* 两阶段封锁协议
* 死锁意义与处理，预防
* 什么是饿死，

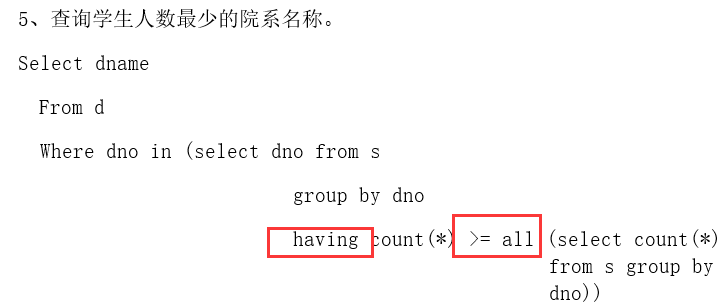


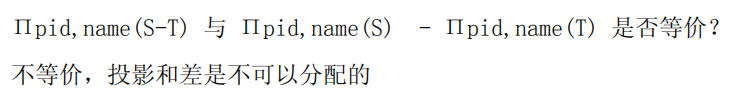
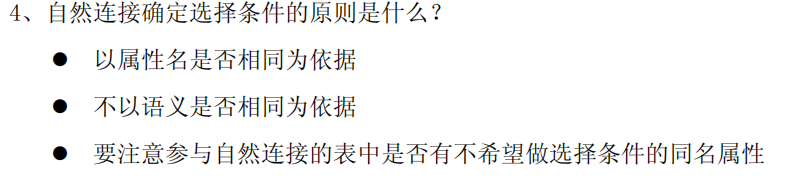
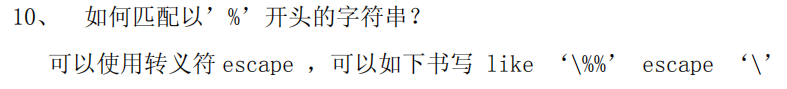


* 结果写a.sno







* 出现同名属性就 r.a把表名加上
* 
* 
* 
* sql 语句中各子句的执行顺序是什么？

如下所示，前边的标号表示执行顺序：

⑤select s.sno,sname,avg(score)

① from s,sc

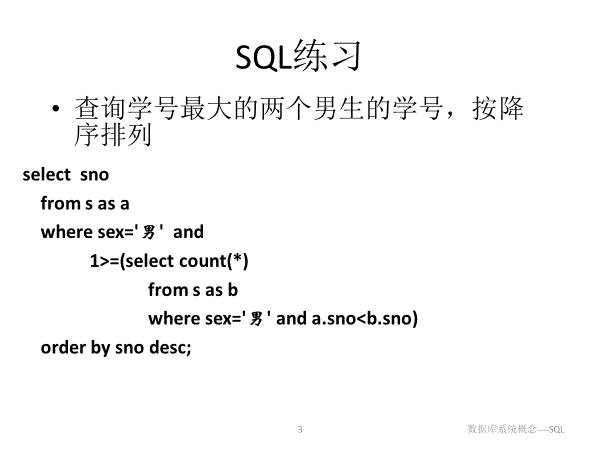
② where s.sno=sc.sno

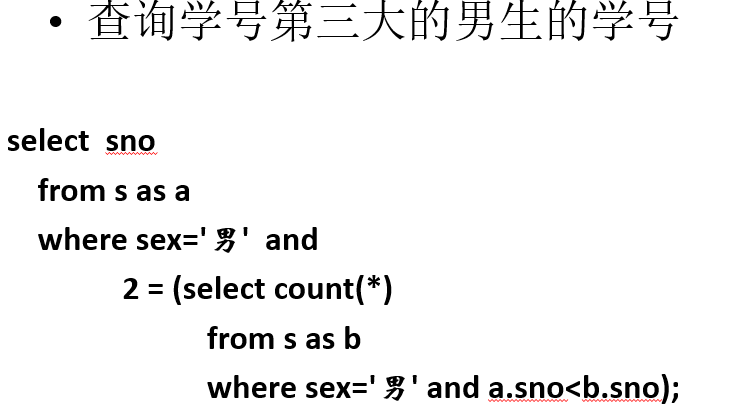
and score>=60

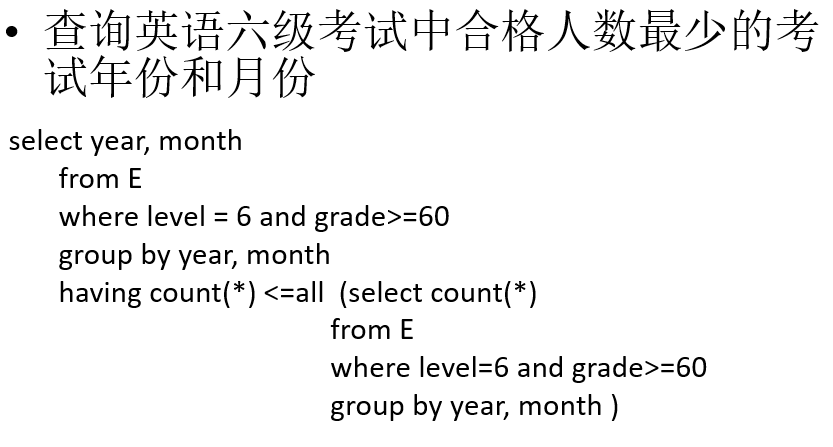
③ group by s.sno,sname

④ having count(\*)>=3

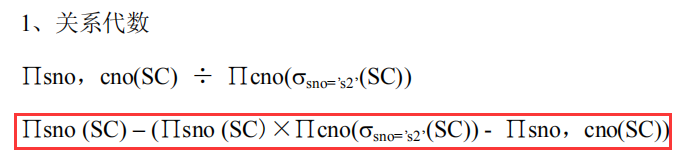
⑥ order by s.sno;



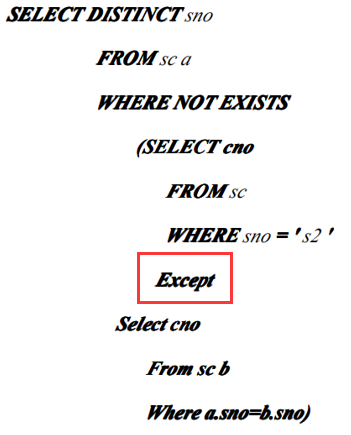


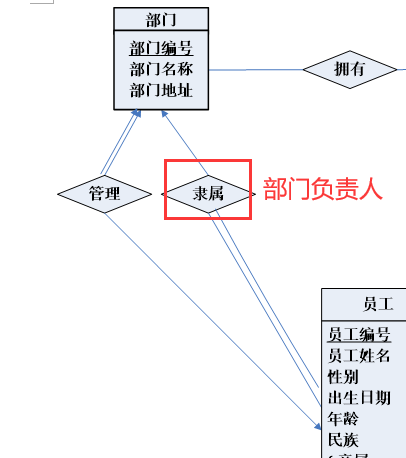
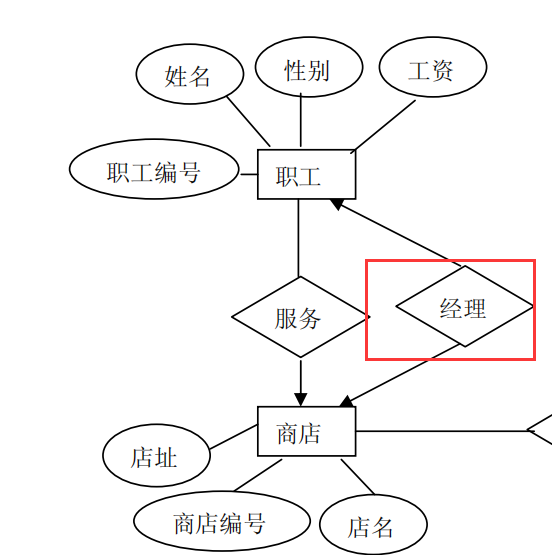


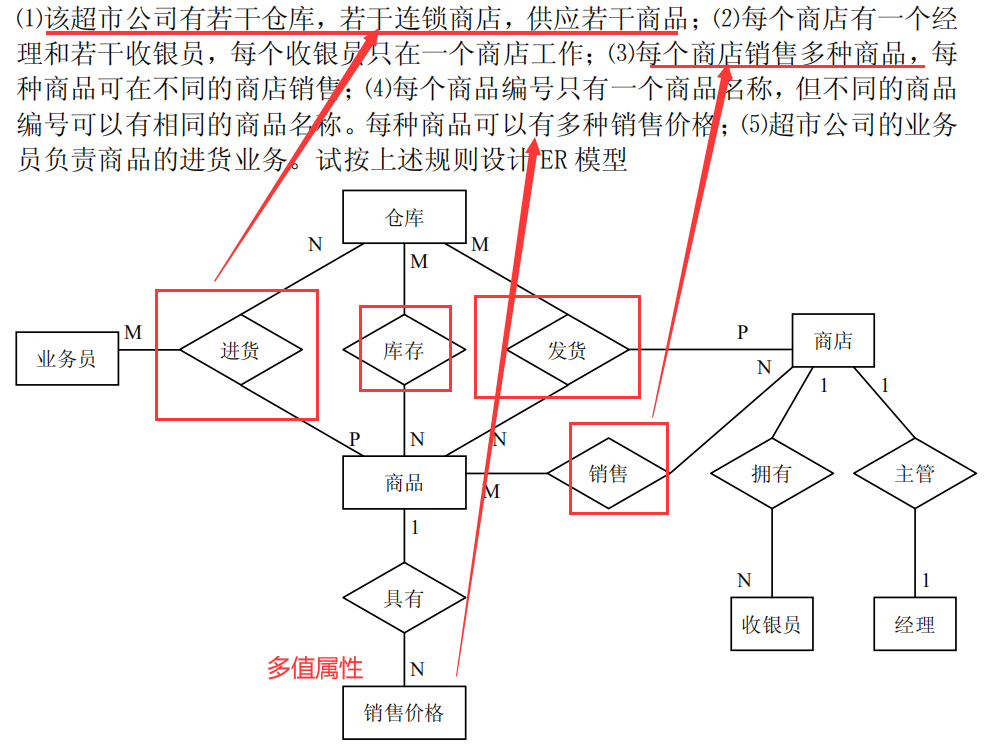
* 查询选择了学号是 s2 学生选修的所有课程的学生学号

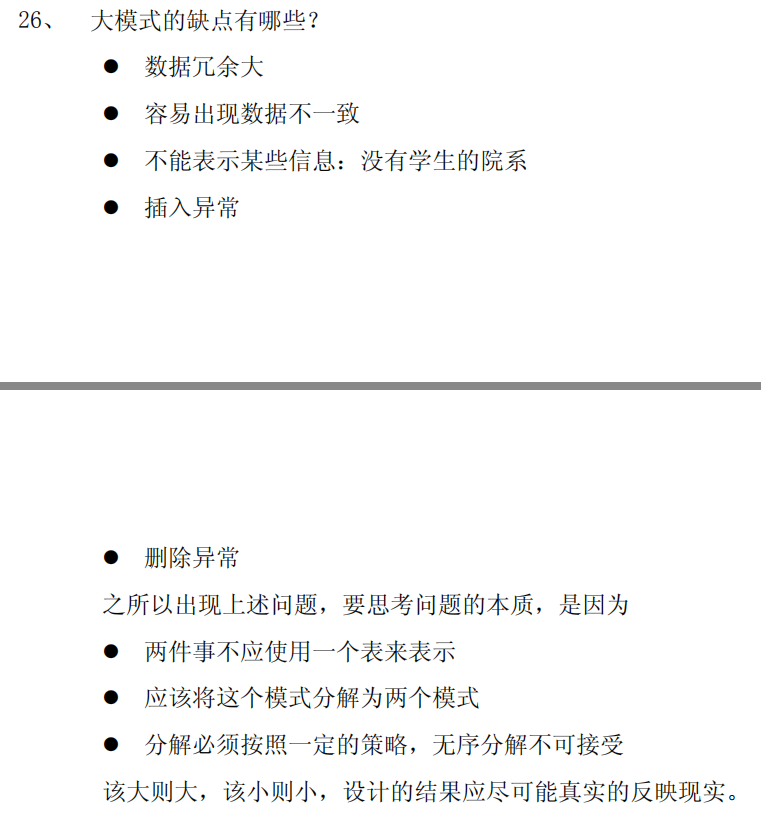


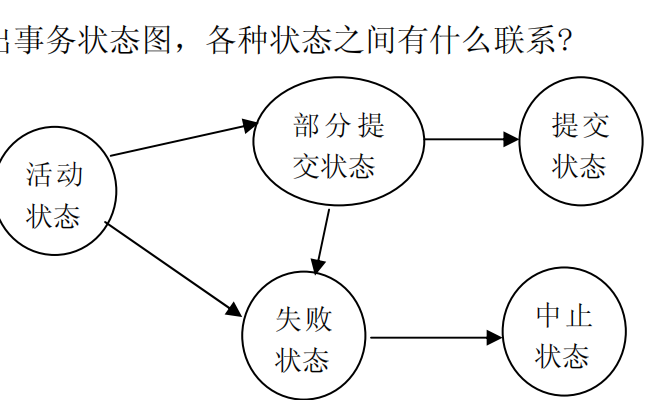


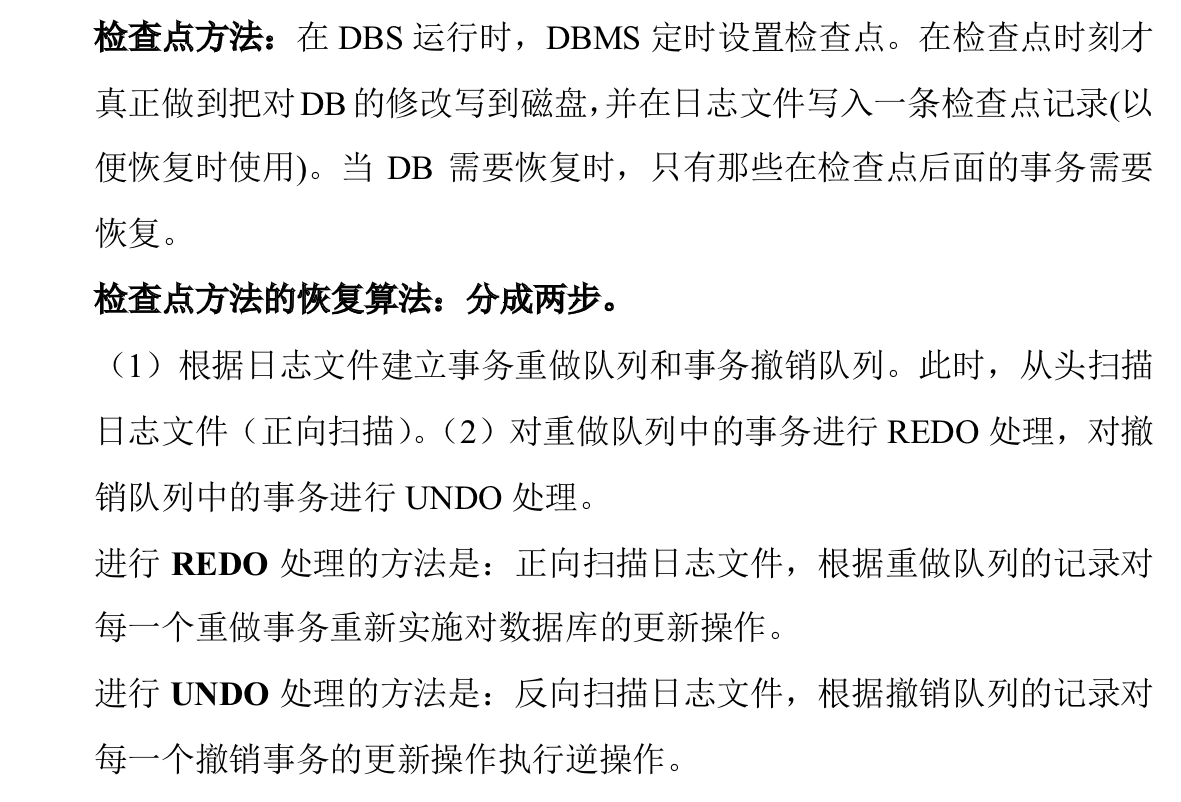










题里补充点

* 关系模式和关系实例：

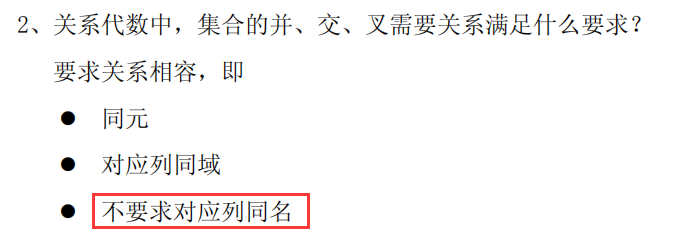
关系模式由属性序列和各属性对应的域组成。关系实例表示一个关系的特定实例，也就是所包含的一组特定的行。模式是相对稳定的，而实例是相对变动的。模式描述的是数据的结构及其联系，实例反映的是数据库某一时刻的状态

* 文件系统阶段和数据库系统阶段对比

1. 这两个系统都包含一组数据和一组访问这些数据的程序。数据库管理系统协调对数据的物理和逻辑访问，而文件处理系统只协调对数据的物理访问（数据独立性不够）；
2. （数据不是依赖处理过程的附属品，而是现实世界中独立存在的对象）数据库管理系统通过确保一个物理数据片段对所有有权访问它的程序都可用，从而减少数据重复，而由文件处理系统中的一个程序编写的数据可能不会被另一个程序读取（数据冗余，不一致，孤立）；
3. 数据库管理系统的设计是为了允许灵活地访问数据（DBMS统一存取，维护数据语义和结构，自动检查完整性约束）。，而文件处理系统的设计是为了允许预先确定的数据访问(如，被编译的程序)；
4. 并发访问问题：设计了一个数据库管理系统来协调多个用户同时访问相同的数据。文件处理系统通常设计成允许一个或多个程序同时访问不同的数据文件。在文件处理系统中，只有当两个程序对文件具有只读访问权限时，两个程序才能同时访问文件
5. 安全性约束，文件系统难以实现
6. 区别：

|  |  |
| --- | --- |
| 文件系统 | 数据库系统 |
| 程序和数据有一定的联系 | 程序和数据分离 |
| 用操作系统中的存取方法对数据进行管理 | 用DBMS对数据进行管理 |
| 实现以文件为单位的数据共享 | 实现以记录和字段为单位的数据共享 |
| 用文件将数据长期保存在外存上 | 用数据库统一存储数据 |

联系：均为数据组织的管理技术，均有数据管理软件管理数据，程序与数据之间用存取方法进行转换，数据库系统是基于文件系统的。



* 2． 数据库设计分哪几个阶段？

数据库设计分以下六个阶段：

（1）需求分析

（2）概念结构设计

（3）逻辑结构设计

（4）物理结构设计

（5）数据库实施

（6）数据库运行和维护

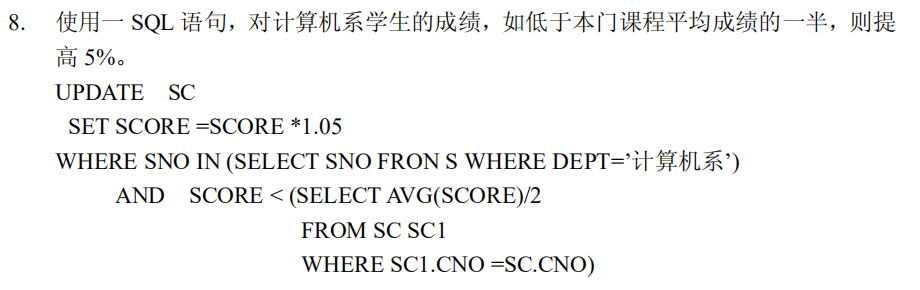
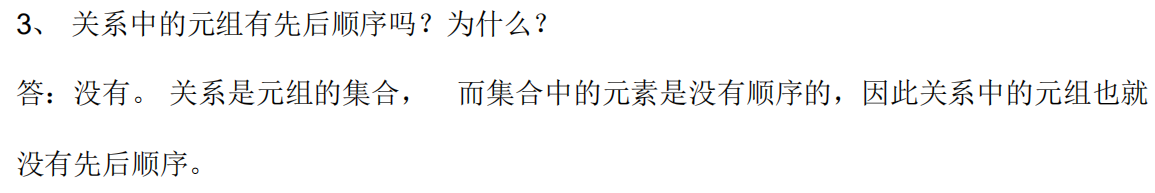
* 简述串行调度与可串行化调度的区别：

串行调度是多个事务按照一定的次序依次执行；在某一时刻只有一个事务在执行。 而可串行化调度是并发调度中的一个概念。在并发调度中,某一时刻有多个事务同时被处理。如果某事务集的一个并发调度的结果与某个串行调度等价，则称此并 发调度是是可串行化调度。

* 在嵌入式 SQL 中，什么情况下的 DML 语句不必涉及到游标操作？

INSERT、DELETE 和 UPDATE 语句；

对于 SELECT 语句，如果已知查询结果肯定是单值时。

* 
* 
* 简述何时需要建立聚簇索引，以及它的优缺点。

如果经常需要按照某搜索码（在文件中查找记录的属性或属性集）做查找或者排序操作，可在该搜索码上建立聚簇索引，使得数据文件的记录顺序按照搜索码的顺序排序。

优点：利用搜索码上的聚簇索引，可以基于搜索码属性做高效的查找和排序操作；聚簇索引可以采用稀疏方式存储，节省索引的存储成本。

缺点：聚簇索引要求数据文件的记录顺序按照搜索码的顺序排序，因而在一个关系上只能建立一个聚簇索引。但对数据新增/修改/删除的影响比较大。如果基于非聚簇索引（辅助索引）的搜索码进行查找操作，成本会相对高一些。

* 2.索引：

索引是一个单独的、物理的数据库结构，它是某个表中一列或若干列值的集合和相应的指向表中物理标识这些值的数据页的逻辑指针清单。 索引提供指向存储在表的指定列中的数据值的指针，然后根据指定的排序顺序对这些指针排序。

索引优点：

第一， 通过创建唯一性索引，可以保证数据库表中每一行数据的唯一性。

第二， 可以大大加快数据的检索速度，这也是创建索引的最主要的原因。

第三， 可以加速表和表之间的连接，特别是在实现数据的参考完整性方面特别有意义。

第四， 在使用分组和排序子句进行数据检索时，同样可以显著减少查询中分组和排序的时间。

第五， 通过使用索引，可以在查询的过程中，使用优化隐藏器，提高系统的性能。索引

索引类型：

顺序与散列：

顺序索引：基于值的顺序排序

散列索引：基于将值平均分布到若干散列桶中。一个值所属的散列桶是由一个散列函数决定的。

聚集索引与非聚集索引：

聚集索引：数据表的物理顺序与索引的制定顺序相同的索引。

非聚集索引：数据表的物理顺序与索引指定的顺序不同的索引。

稠密索引与稀疏索引：

稠密索引：每个搜索码的值都有一个索引项

稀疏索引：只为搜索码的某些值建立索引项，只有当关系按搜索码排列顺序存储时才能使用稀疏索引，也就是只有当索引是聚集索引时才能使用稀疏索引

3.索引顺序文件的缺点及B+树索引优缺点

(1) 索引顺序文件的缺点（主要在结构方面）

①随着文件的增大，由于许多溢出块会被创建，索引查找性能和数据顺序扫描性能都会下降。

②插入和删除时，频繁重组整个文件。

(2) B+树索引文件的优点

①在数据插入和删除时，能够通过小的自动调整来保持平衡。

②不需要重组文件来维持性能。

(3) B+树索引文件的缺点

增加文件插入和删除的时间开销，同时会增加空间开销。这是因为插入和删除可能会引发 B+树的调整，并且树形结构比线性存储需要更大的空间。

总结：相较于顺序存储时频繁地调整整个文件，B+树只需要在局部调整以维持平衡，并且 B+树查找每个叶节点的性能非常稳定。所以我们可以接受一定的时间和空间开销，使用这种数据结构。

* .证明为什么两阶段封锁协议能够保证事务集合的可串行化。

采用反证法，假定两阶段封锁协议不能保证可串行化，则存在一个遵循两阶段封锁协议的事务集合T0, T1, …, Tn-1产生了一个非串行的调度，那么在其优先图中会有一个环。不失一般性，假定优先图中的环是T0 T1 T2… Tn-1 T0，让ai对应事务Ti获得到最后一个锁的时间（即事务Ti的封锁点）。因而对于所有的事务Ti<Tj，都有ai < aj，对于这个环，我们就有 a0 a1 a2 … an-1 a0，存在矛盾，因此假设不成立。

所以，两阶段封锁协议能够保证事务集合的可串行化

* 12.2.3 并行与串行

(1) 基本比较

①并行事务会破坏数据库的一致性。

②串行事务效率低。

(2) 并行的优点

①一个事务由不同的步骤组成，所涉及的系统资源也不同。这些步骤可以

并发执行，以提高系统的吞吐量(throughput)。

②系统中存在着周期不等的各种事务，串行会导致难于预测的延迟。如果

各个事务所涉及的是数据库的不同部分，采用并行会减少平均响应时间

(average response time)。

(3) 核心问题

在保证一致性的前提下最大限度地提高并发度。

(4) 并发操作面临的问题

①丢失修改（lost update）

丢失修改是指事务 1 与事务 2 从数据库中读入同一数据并修改，事务 2 的

提交结果破坏了事务 1 提交的结果，导致事务 1 的修改被丢失。

②不可重复读（non-repeatable read）

不可重复读是指事务 1 读取数据后，事务 2 执行更新操作，使事务 1 无法

再现前一次读

取结果。

③读“脏”数据（dirty read）

事务 1 修改某一数据，并将其写回磁盘，事务 2 读取同一数据后，事务 1

由于某种原因被撤消，这时事务 1 已修改过的数据恢复原值，事务 2 读到的数

据就与数据库中的数据不一致，是不正确的数据，又称为“脏”数据。

注：并行操作面临的问题将由并发控制系统解决。

(5) 并行调度的原则

并行调度应该在某种意义上等价于一个串行调度。

①数据库系统的调度应该保证任何调度执行后数据库总处于一致状态。

②通过保证任何调度执行的效果与没有并发执行的调度执行效果一样，可

以保证数据库的一致性。

* 给定关系模式R，其上的函数依赖集F，及其分解{R1，R2}，证明如果R1∩R2→R1属于F+，则R1和R2是R的无损分解。

答：假设 ∀r∈ R，分解得到r1, r2，只需证明r = r1|><| r2。

容易知道，∀t∈r，t∈ r1|><| r2 ，所以 r ⊆ r1|><| r2。

令α = R1∩R2 ，β = R1 - α , γ = R2 - α

∀t ∈ r1 |><| r2 , 得 t[αβ] ∈ r1 , t[αγ] ∈ r2 ；

设t[αβ]在r里面对应元组为t1，则t1[α]= t[α]，t1[β]=t[β]；

设t[αγ] 在r里面对应元组为t2，则t2 [α]= t[α]，t2[γ]=t[γ]；

由α→R1 得α→β，又因为t1[α] = t2[α] = t[α]，得t1 [β] = t2 [β]，所以t2[β] = t[β]，所以t2 = t，所以t∈ r；

得 r1|><| r2 ⊆ r。

因此r = r1|><| r2得证。

* 查询处理的基本步骤

1，语法分析与翻译

把查询语句翻译成系统的内部表示形式，也就是翻译成关系代数

语法分析器检查语法，验证关系

2，查询优化: 在所有等效执行计划中选择具有最小查询执行代价的计划 （用于执行一个查询的原语操作序列称为查询执行计划）

使用来自数据库目录的统计信息来评估代价

例如，每个关系中的元组数、元组大小等

3，执行

查询执行引擎接收一个查询执行计划，执行该计划并把结果返回给查询