复习练习题

Karry
不打无准备之仗

1 选择题 （5 * 2 = 10 分）

1. 数据库中产生数据不一致的根本原因是数据冗余
2. 并发控制是为了确保事务的隔离性
3. 概念模型是实现现实世界的第一层抽象，这一类模型中最著名的模型是实体-联系模型
4. 五种基本关系代数运算是并、差、选择、投影、笛卡尔积，因为其他的各种运算功能都能由着五种运算实现
5. 关系数据库中必须满足：每一个属性都是不可分解的
6. 一个关系模式如果属于 4NF，则一定属于 BCNF，反之不成立
7. 关系数据规范化是为了解决关系数据库中插入异常、删除异常和数据冗余问题而引出的。
8. 数据库的故障恢复中，不需要做 redo 操作的是事务故障
9. BCNF 并不能消除所有的冗余

2 简答题 （4 * 5 = 20 分）

1. 什么是数据独立性？数据库系统如何提供数据独立性？

   数据独立性是指应用程序和数据之间相互独立、不受影响，即数据结构的修改不会引起应用程序的修改。

   数据独立性包括物理数据独立性和逻辑数据独立性，物理数据独立性是指数据库物理结构改变时不必修改现有的应用程序（数据库存储结构的改变会引起内模式的改变，通过改变模式与内模式之间的映像，使数据库的模式保持不变，从而不必修改应用程序），逻辑数据独立性是指数据库逻辑结构改变不用改变应用程序。

   数据独立性由数据库系统三级模式之间的两层映像功能来保证的。

2. 什么是数据库系统的三级模式结构？他是如何保证数据独立性的？

   三级模式结构为：数据库的外模式、模式、内模式

   数据按外模式的描述提供用户，按内模式的描述存储在硬盘上。模式介于内、外模式之间，既不涉及外部的访问，也不涉及内部的存储，从而起到隔离作用，有利于保持数据的独立性

3. 什么是可串行化调度？如何保证并发调度的可串行性？

在并发执行的情况下，通过保证所执行的任何调度的效果都与没有任何并发执行的调度效果一样来确保数据的一致性，调度在某种意义上等价于一个串行调度，这种调度被称为可串行化调度。

两阶段封锁协议是保证可串行化调度的一种方式。

4. 试述“串行调度”与“可串行化调度”的区别

事务的执行次序称为调度，如果多个事务一次执行，则称为事务的串行调度。如果利用分时的方法，同时处理多个事务，则称为事务的并发调度。
如果一个并发调度的结果与某一个串行调度的执行结果等价，那么这个并发调度称为可串行化调度。

5. 试述两阶段封锁协议的概念

该协议要求每个事务分两个阶段提出加锁和解锁申请，分别是增长阶段（一个事务可以获得锁，但不能释放任何锁）与缩减阶段（一个事务可以释放锁，但不能获得任何新锁），起初一个事务处于增长阶段，不断加新锁，一旦该事务释放了一个锁，他就进入了缩减阶段并且它不能再发出加锁请求。

6. 简述两阶段封锁协议的内容

所有事务必须分两个阶段对数据项加锁和解锁：
（1）对任何数据进行读和写操作之前，首先要申请并获得对该数据的封锁。
（2）在释放一个封锁后，事务不再申请和获得任何其他的封锁。

7. 数据库中日志文件有何作用？利用日志文件如何进行系统故障的恢复？

【日志作用】日志文件是用来记录事务对数据库更新操作的文件，可以用来进行事务故障的恢复和系统故障的恢复，并协助后备副本进行介质故障的恢复

【恢复方法】系统发生故障后，可以根据日志内容对事务执行情况进行判断，进而对那些尚未完成的事务通过 undo 进行撤销处理，对那些已经完成的事务通过 redo 进行重做处理。即正向扫描日志文件，对已提交的事务标记进入 REDO 队列，未完成的事务标记进入 UNDO 队列。

8. 关系数据库的参照完整性规则

引用关系中的任意元组在指定属性上出现的取值也必然出现在被引用关系中至少一个元组的指定属性上。

9. 数据库系统的故障有哪些类型并举例说明

1） 事务故障：

- 逻辑错误：事务由于某些内部情况而无法继续其正常执行，这样的内部情况诸如非法输入、找不到数据、溢出或超出资源限制
- 系统错误：系统进入一种不良状态，其结果是事务无法继续其正常执行，但该事务可以在之后的某个时间重新执行

2） 系统故障：硬件故障，或者是数据库软件或操作系统的漏洞，导致易失性存储器内容的丢失，并使得事务处理停止。但非易失性存储器内容完好无损且没被破坏。

3） 介质故障：在数据传输操作中由于磁头损坏或故障造成磁盘块上的内容丢失。

10. 数据库系统如何进行系统故障的恢复？

系统重新启动时，撤销所有未完成（非正常终止）的事务，并重做所有已提交的事务

11. 数据库备份中，什么是动态转储，为何动态转储需要日志文件？

【定义】转储即数据库管理员定期将整个数据库复制到存储介质中保存起来的过程，动态转储是指在转储期间允许对数据库进行存取或修改。

【原因】由于转储时有用户事务执行，动态转储结束后，后援副本上的数据不能保证完全正确有效，因此必须把转储期间各事务对数据库的修改活动记录下来，建立日志文件，这样后援副本加上日志文件就能把数据库恢复到某一个时刻的正确状态。

12. 所有的视图是否都可以更新？为什么？

更新视图是指通过试图来插入、删除和修改数据，由于视图是不实际存储数据的虚表，因此对视图的更新最终要转换为对基本表的更新。为了防止用户通过视图对数据进行插入、删除和修改甚至对不属于视图范围的基本表数据进行操作，所以一些相关措施使得不是所有的视图都可以进行更新。在SQL中视图定义时，可以加上 `WITH CHECK OPTION` 子句，这样在视图上增、删、改数据时，DBMS 就会检查视图定义中的条件，如果不满足四个条件，则拒绝执行操作。

四个条件分别是什么

如果不满足就更新会出现什么问题？

13. 为什么将 SQL 中的视图称为“虚表”？

在 SQL 中创建一个视图时，系统只是将视图的定义存放在数据字典中，并不存储视图对应的数据。只有到用户使用视图时才会去求对应的数据，因此将视图称为虚表。这样处理的目的是为了节约空间，因为视图中对应的数据都是从相应的基本表中获得的。

14. 为什么要对关系代数表达式进行优化？

关系代数表达式由关系代数操作组合而成，其中笛卡尔积和连接操作最费时，如果DBMS直接按照用户书写的表达式顺序执行，必将花费很多时间，同时生成的大量中间结果会占用内存，效率极低。

因此需要在查询执行之前需要 DBMS 查询子系统先对关系代数表达式进行优化，尽可能先执行选择和投影操作，那么再进行笛卡尔积或者连接时就可以减少中间结果节约内存并且节约时间。

15. 列出三条查询优化的启发式规则：

3 条启发式优化规则是：

1） 尽可能早地执行选择运算

2） 尽可能早地执行投影运算

3） 将笛卡尔积与附近地一连串选择合并形成连接运算

16. 试述等值连接与自然连接的区别

1） 自然连接一定时等值连接，但等值连接不一定是自然连接，因为自然连接要求相等的分量必须是公共属性，而等值连接要求的分量不一定是公共属性。

2） 等值连接不把重复属性去掉，自然连接会去掉重复属性

17. 数据库设计的基本步骤按顺序有哪些？在哪个阶段进行关系数据库的基本表设计？

1） 系统需求设计

2） 概念设计

3） 逻辑设计

4） 物理设计

5） 数据库实施

6） 数据库运行和维护

在逻辑设计阶段，将概念设计的 E-R 模型转化为基本表

18. 数据库管理系统的主要功能有哪些？

1) 数据库运行管理

2) 数据库的建立和维护

3) 数据定义功能

4) 数据操纵功能

5) 数据组织、存储和管理

19. 关系数据库的查询处理依次分为那几个阶段？

1) 查询分析

2) 查询检查

3) 查询优化

4) 查询处理

20. 简述相关子查询的一般求解过程

首先，外层关系中的每一个元组都被带入内层；

内层得到结果，外层的 `WHERE` 进行条件判断，如果为真，当前的元组被选出，否则不被选出；

重复直到外层没有元组为止

3 分析题 （2 * 10 = 20 分）

3.1 关系模式分析

3.1.1 根据约束条件写函数依赖，并且求候选码

求候选码的算法

一个例子： $R(A,B,C,D); F=\{B\to D, D\to B, AB\to C\}$

先看属性在左右两边出现的情况

- `L` - 仅在左边出现
- `LR` - 左右两边都有
- `R` - 仅在右边出现

`L` : {A} 首先怀疑 `A` 是候选码，但是发现什么都推不出来

`LR` : {B, D} 那么怀疑 {A, B} {A, D} 是候选码，发现果然都是，就不再继续了

因此候选码就是 {A,B} {A, D}

例子

1. 项目（项目名，项目组长，指导老师，该老师指导的项目数）

   假定：一个项目一位组长，一个项目只有一位指导老师

   Step 1. 写函数依赖

   项目名 -> 项目组长

   项目名 -> 指导老师

   指导老师 -> 该老师指导的项目数

   Step 2. 依靠上述算法求出候选码

   {项目名}

2. R（职工编号，日期，日营业额，所属部门名，部门经理）

   约束条件：每个职工每天只有一个营业额，每个职工只属于一个部门，每个部门只有一个经理

   Step 1. 函数依赖

   职工编号，日期 -> 日营业额

   职工编号 -> 所属部门名

   所属部门名 -> 部门经理

   Step 2. 依靠上述算法求出候选码

   {职工编号，日期}

   Step 3. 由于存在函数依赖 {职工编号 -> 所属部门名} 即非码属性部分依赖于候选码，所以是 2NF

   Step 4. 分解成 3NF

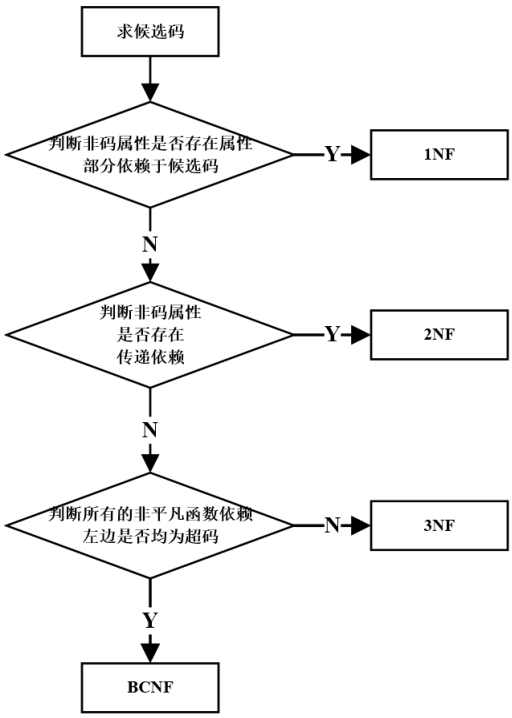
   R1(职工编号，日期，日营业额度)

   R2(职工编号，所属部门名)

   R3(所属部门名，部门经理)

3.1.2 判断范式等级

范式等级判断算法



例子

1. 项目（项目名，项目组长，指导老师，该老师指导的项目数）

假定：一个项目一位组长，一个项目只有一位指导老师

明显存在非码属性，传递依赖，所以是 2NF

分解后：

R1 项目组（项目名， 项目组长）

R2 项目指导（项目名， 指导老师）

R3 老师指导项目（指导老师，该老师指导的项目数）

2. 职工（职工号，姓名，基本工资，工作部门）

【函数依赖】

职工号 -> 姓名

职工号 -> 基本工资

职工号 -> 工作部门

所以是 BCNF 不需要分解

3. R（A， B，C，D），F = {D -> B, CD -> A}

CD 是候选码

存在非码属性部分依赖，所以是 1NF

分解后：

R1(DB)

R2(CDA)

4. R（A，B，C），F = {C -> A，AB -> C}

{AB} {CB} 是候选码

C -> A 存在，因此肯定不是 BCNF 又因为 A - C = A 属于候选码

所以是 3NF

分解成 BCNF

R1(A, C)

R2(B, C)

3.1.3 范式分解

一个例子：

$R(A,B,C,D,E,F); F = \{AE \to F, A \to B, BC \to D, CD \to A, CE \to F\}$

3.1.3.1 三范式分解

Step 1. 求候选码

先看属性在左右两边出现的情况

L - 仅在左边出现 | LR - 左右两边都有 | R - 仅在右边出现

L : {C, E}

首先怀疑CE 是候选码，但是发现不行

LR : {A,B,D}

从 LR 中选出一个和 CE 组合

ACE => 可以推出

BCE => 可以推出

DCE => 可以推出

一共有三个候选键

R : F

因此候选码就是 ACE BCE DCE

Step 2. 求函数依赖 F 的正则覆盖 F_c，发现该函数依赖 F 本身就是一个正则覆盖

Step 3. 把 F_c 都写成关系，并检查是否存在候选码，如果没有候选码，就补一个候选码作为一个关系

$$\rho = \{R_1(AEF), R_2(AB), R_3(BCD), R_4(CDA), R_5(CEF)\}$$

发现不存在候选码，所以补一个候选码作为关系即可

$$\rho^{3NF} = \{R_1(AEF), R_2(AB), R_3(BCD), R_4(CDA), R_5(CEF), R_6(ACE)\}$$

Step 4. 如果其中有关系包含在另外的关系当中，那么就要删去该关系。上述给出的分解没有被包含在其中的关系。

至此分解完毕！

3.1.3.2 BCNF 范式

Step 1. 求候选码 和上面相同

Step 2. 求函数依赖 F 的正则覆盖 F_c，发现该函数依赖 F 本身就是一个正则覆盖

Step 3. 逐层分解

R(ABCDEF) F{AE->F, A->B, BC->D, CD->A, CE->F}

1.

因为 AEF 不是候选码，所以构成 R1(AEF)

同时R -> (ABCDE) F{A->B, BC->D, CD->A} 所有含 F 的都没了，因为 F 可以被推出

2.

因为 AB 不是候选码，所以构成 R2(AB)

同上 R -> (ACDE) F{CD -> A}

3.

因为 CDA 不是候选码 所以构成 R3(CDA)

同上 R -> (CDE) F{无}

R4 = (CDE) 肯定是一个候选码

综上得出：R 的分解 R1 R2 R3 R4

3.2 加锁，减锁的写法

1. 基本上就是考察两道封锁协议的过程，注意加锁类型，千万不要给要 write 的数据加上了共享锁
2.

例子

1. 考虑如下两个事务

```
T1: read(A)
    read(B)
    if A = 0 then B:B+1
    write(B)
T2: read(B)
    read(A)
    if B = 0 then A:=A+1
    write(A)
```

按照两端封锁协议分别加锁和解锁，指出并发执行时会不会死锁

```
T1: lock-S(A)  // 也可以 lock-X(A)
    read(A)
    lock-X(B)
    read(B)
    if A = 0 then B:=B+1
```

```
    write(B)
    unlock(A)
    unlock(B)
T2: lock-S(B) // 也可以 lock-X(B)
    read(B)
    lock-X(A)
    read(A)
    if B = 0 then A:=A=1
    write(A)
    unlock(B)
    unlock(A)
```

如果 T1 先给 A 上了锁 还没给 B 上锁时间就开始执行 T2 并且直到 T2 执行到 read(A) 后，T1 执行给 B 上锁，这样就死锁了，T2 等待 T1 解锁 A， T1 等待 T2 解锁 B

2.

4 查询题：（5 * 6 = 30 分）

4.1 关系代数

4.2 SQL 语言

1. 查询选修了某一位同学选修的所有课程的同学

```
not exist (A except B)
```

2. 查询每一位读者的借书数量

```
select count(book_ID)
from reader natural join borrow,
group by reader_ID
```

3. 查询所有学生都选择的课程的课程号

```
逆向思维，查询所有学生 - 查询某一个课程其所有的选修者
如果不为空说明没有被所有学生都选择
```

4. 找出所有其报酬高于所在项目平均报酬的员工所在的项目号，姓名和报酬

```
with p_avgpay(pNo, avg_pay) as
    select pNo, avg(pay) as avg_pay
    from pe
```

```
    group by pNo
''' 先找出所有项目的平均工资，再将平均工资表和参与表自然连接，然后找到最后的结果 '''
select eName, pNo, pay
from pe natural join p_avgpay
where pe.pay > p_avgpay.avg_pay

select eName, pNo, pay
from pe as A
where A.pay > (select avg(pay)
               from pe as B
               where A.pNo=B.pNo)
```

5 设计题（2||3 *5=10 || 15分）

5.1 画E-R图：根据给定的语义（要求）画出E-R图 (5分)

注意点：

1. 关系集的属性用虚线
2. 注意基数和参与性限制

5.2 关系模式转化：该E-R模型转换为满足3NF的关系模型，并指出主码和外码(5 分)

Step 1. 先找总关系的候选码

Step 2. 将每一个框 不管是实体集还是联系集，都写成一个关系

Step 3. 看包含候选码否，不包含再加一个

5.3 查询：基于上述的关系模式，用关系代数写出指定查询：(5分)

6 优化题

6.1 画出初始语法树

1. Step 1 写出关系代数表达式

2. Step 2 画树，

6.2 画出优化语法树

1. 投影早做
2. 选择早做