数据库系统原理复习笔记

绪论

1. 明晰几个概念：

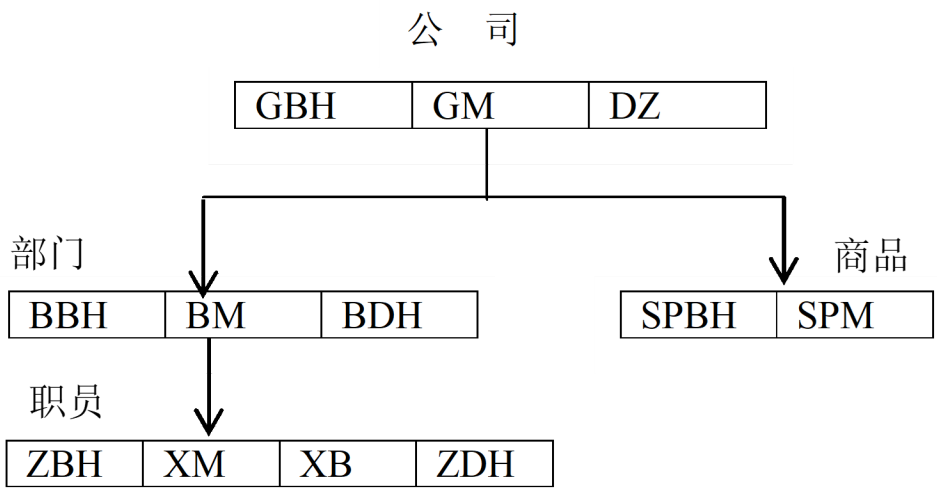
- 数据（Data）
- 数据库（**DataBase，DB**）：**长期存储**在计算机内、**有组织**、**可共享**的大量数据集合，其中的数据具有**较小冗余度、较高数据独立性、易扩展性。**
- 数据库管理系统（**DBMS**）
- 数据库系统（**DBS**）

2.数据库的发展历史

- 人工管理阶段： 无磁盘，性能差劲
- 文件系统阶段： 有磁盘，可长期存储
- 数据库系统阶段

3.数据模型

- 概念模型
  - 实体、属性、码（属性或由多个属性组成的属性集，如学号是学生实体的码）、实体型（实体名及其属性集合）、实体集（如全体学生）、联系（**一对一、一对多、多对多**）
- 逻辑模型
  - **层次模型**



    - 特征
      - **仅根节点无双亲结点**
      - **其余子女结点有且只有一个双亲结点（子女结点与双亲结点之间的联系唯一）**
    - 操作：增加、删除、修改、查询
    - 优点：简单、速度快、容易表示一对多联系
    - 缺点：**不能直接表示多对多联系**、插入和删除限制多、查找子女需要经过双亲
    - 约束：无双亲不能插入子女结点的值；删除双亲结点的值，则子女结点的值一同删去
  - **网状模型**

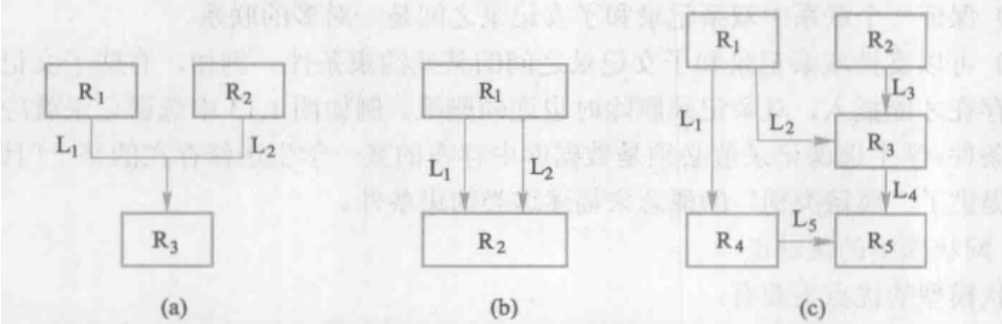


图 1.12　网状模型的例子

- 特征
  - 允许多个结点无双亲结点
  - 一个结点可以有多个双亲结点（如图a）
  - 两个结点之间可以有多种联系（复合联系，如图b）
- 操作：增加、删除、修改、查询
- 优点：可以直接表示多对多联系、存取效率高
- 缺点：结构复杂、一次存取一个记录值、应用程序与数据结构互相依赖
- 约束：插入不受限制；删去双亲结点值，子女结点不受影响

- **关系模型**：关系对应一张表，一行是一个元组，一列是一个属性，表中一个属性组可以唯一确定一个元组（码），关系模式（如学生（学号，姓名，年龄，性别，系名，年级）是一个关系模式）
  - 存取路径对用户透明，具有极高的数据独立性和安全保密性

4.三级模式和两级映像

- 外模式：数据库用户能看见和使用的局部数据的逻辑结构和特性的描述，是模式的子集
- 模式：数据库中全体数据的逻辑结构和特性的描述
- 内模式：对数据的物理结构和存储方式的描述
- 两级映象保证了数据库系统中数据具有较高的逻辑独立性和物理独立性

关系

1.明晰关系的概念（对应绪论里的关系）

- 候选码：最小码，候选码的属性称作主属性，其余的是非主属性或叫做非码属性
- 全码：包括关系模式中的所有属性
- 主码：候选码之一
- 外码：R1的属性子集但不是R1的码，是R2的码，则是R1的外码（Student : 被参照关系、Course : 被参照关系、SC : 参照关系，SC中的Sno和Cno是外码, 注意不是(Sno, Cno)）

关系代数

表 2.4 关系代数运算符

运算符		含义
集合运算符	∪	并
	−	差
	∩	交
	×	笛卡儿积
专门的关系运算符	σ	选择
	Π	投影
	⋈	连接
	÷	除

- $R-S$：由属于R但不属于S的元组组成
- $\sigma_F(R)$：从关系表R中选择使得逻辑表达式F为真的元组(对应教材例2.4)
  - $\sigma_{Sdept='IS'}(Student)$: 查询信息系全体学生
  - $\sigma_{Grade>90 \vee Grade<45}(SC)$：查询成绩大于90分或小于45分的选课记录
- $\Pi_A(R)$：从R中选出若干列属性列, A为R中的若干列属性名
  - $\Pi_{Sname,Sdept}(Student)$：查询学生的姓名和所在院系
- $R \bowtie S$：自然连接（特殊的等值连接），结果中去掉重复属性列

书上例子：

［例 2.8］ 设图 2.7(a)和(b)分别为关系 R 和关系 S，图 2.7(c)为非等值连接 $R \underset{C<E}{\bowtie} S$ 的结果，图 2.7(d)为等值连接 $R \underset{R.B=S.B}{\bowtie} S$ 的结果，图 2.7(e)为自然连接 $R \bowtie S$ 的结果。

R

A	B	C
a_1	b_1	5
a_1	b_2	6
a_2	b_3	8
a_2	b_4	12

(a) 关系R

S

B	E
b_1	3
b_2	7
b_3	10
b_3	2
b_5	2

(b) 关系S

$R \underset{C<E}{\bowtie} S$

A	$R.B$	C	$S.B$	E
a_1	b_1	5	b_2	7
a_1	b_1	5	b_3	10
a_1	b_2	6	b_2	7
a_1	b_2	6	b_3	10
a_2	b_3	8	b_3	10

(c) 非等值连接

A	$R.B$	C	$S.B$	E
a_1	b_1	5	b_1	3
a_1	b_2	6	b_2	7
a_2	b_3	8	b_3	10
a_2	b_3	8	b_3	2

(d) 等值连接

A	B	C	E
a_1	b_1	5	3
a_1	b_2	6	7
a_2	b_3	8	10
a_2	b_3	8	2

(e) 自然连接

图 2.7 连接运算举例

有多个公共属性的例子：

R

A	B	C
1	2	3
4	5	6
7	8	9
10	11	12

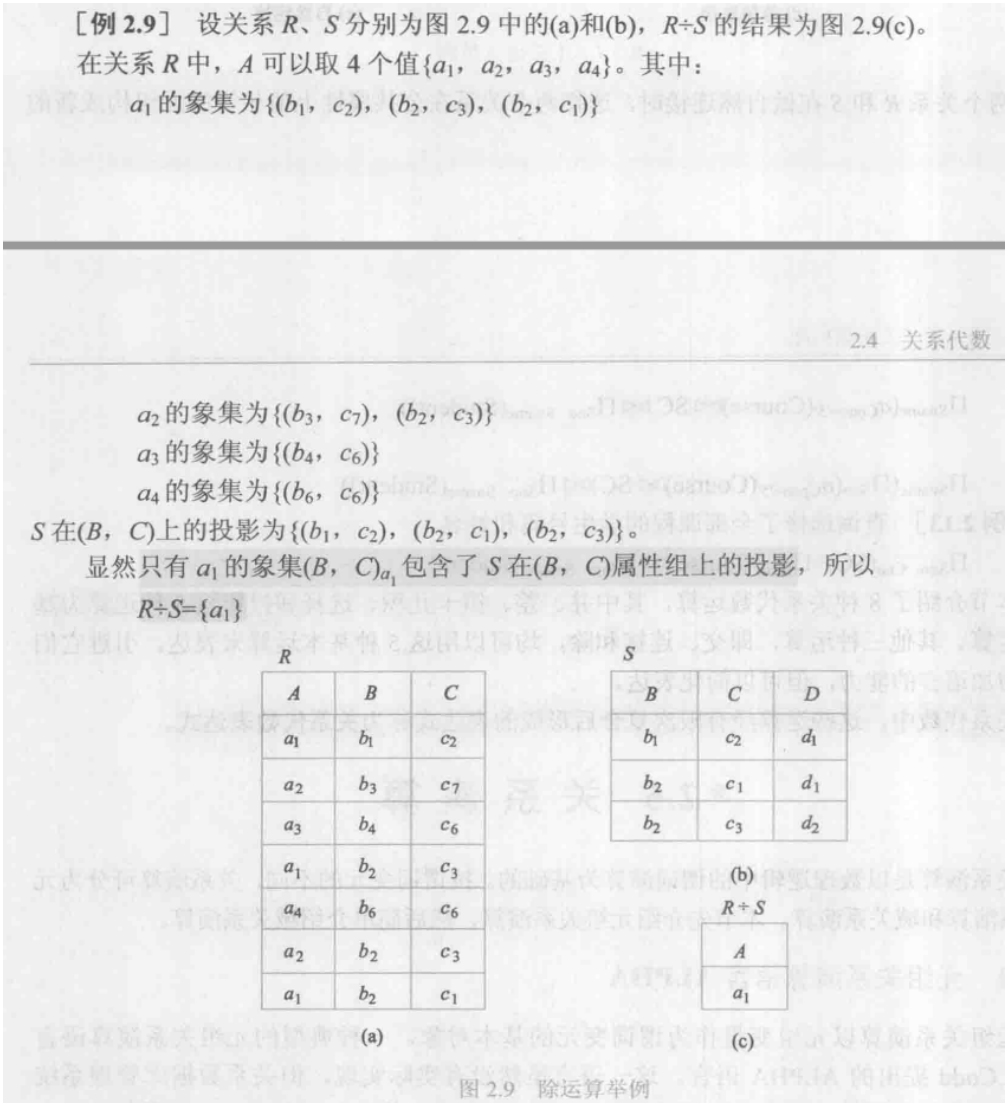
S

B	C	D
2	3	4
3	4	5
5	6	7
7	8	9
8	9	10

则R ⋈ S=

A	B	C	D
1	2	3	4
4	5	6	7
7	8	9	10

- $R÷S$：除运算，我觉得这个运算难以描述，通过例子理解吧。常用于“至少选择多少个”或者“全部”的问题。



［例 2.9］ 设关系 R、S 分别为图 2.9 中的(a)和(b)，$R÷S$ 的结果为图 2.9(c)。

在关系 R 中，A 可以取 4 个值$\{a_1, a_2, a_3, a_4\}$。其中：

a_1 的象集为$\{(b_1, c_2), (b_2, c_3), (b_2, c_1)\}$

2.4 关系代数

a_2 的象集为$\{(b_3, c_7), (b_2, c_3)\}$

a_3 的象集为$\{(b_4, c_6)\}$

a_4 的象集为$\{(b_6, c_6)\}$

S 在(B, C)上的投影为$\{(b_1, c_2), (b_2, c_1), (b_2, c_3)\}$。

显然只有 a_1 的象集$(B, C)_{a_1}$ 包含了 S 在(B, C)属性组上的投影，所以

$R÷S=\{a_1\}$

R

A	B	C
a_1	b_1	c_2
a_2	b_3	c_7
a_3	b_4	c_6
a_1	b_2	c_3
a_4	b_6	c_6
a_2	b_2	c_3
a_1	b_2	c_1

(a)

S

B	C	D
b_1	c_2	d_1
b_2	c_1	d_1
b_2	c_3	d_2

(b)

$R÷S$

A
a_1

(c)

图 2.9 除运算举例

- 综合例题：

［例 2.11］ 查询选修了 2 号课程的学生的学号。

$\Pi_{Sno}(\sigma_{Cno='2'}(SC))=\{201215121,201215122\}$

［例 2.12］ 查询至少选修了一门其直接先行课为 5 号课程的学生姓名。

$$\Pi_{Sname}(\sigma_{Cpno='5'}(Course)\bowtie SC\bowtie \Pi_{Sno,\ Sname}(Student))$$

或

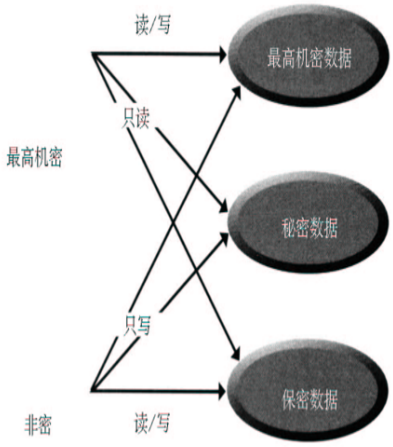
$$\Pi_{Sname}(\Pi_{Sno}(\sigma_{Cpno='5'}(Course)\bowtie SC)\bowtie \Pi_{Sno,\ Sname}(Student))$$

［例 2.13］ 查询选修了全部课程的学生号码和姓名。

$$\Pi_{Sno,\ Cno}(SC)\div\Pi_{Cno}(Course)\bowtie \Pi_{Sno,\ Sname}(Student)$$

数据库安全控制机制

- 用户身份鉴别（口令、生物特征、智能卡）
- 存取访问控制 (防止非授权访问)
  - 自主存取控制 : 不同用户存取权限不同
  - 强制存取控制 : 安全级别较高,按照密级实施权限控制(**读写规则 : 读取保密级别小于或等于自身许可证级别资料; 修改保密级别大于或等于自身许可证等级的资料**)



- 视图机制：对权限低的用户隐藏某些数据
- 审计：将操作自动记录到系统的审计日志中
- 数据加密 : 以密码文的形式存储和运输数据

授权与回收(自主存取控制支持)

```
// grant不允许循环授权
例1:把查询Student表的权限授给用户U1
grant select on table Student to U1;

例2:把Student和Course表的全部操作权限授权给所有用户
```

```
grant all privileges on table Student, Course to public;

例3:把查询Student表和修改学生学号的权限授给用户U4
grant update(Sno), select on table Student to U4;

例4:把对表SC的insert权限授给用户U5,并允许U5将此权限再授给其他用户
grant insert on table SC to U5 with grant option;

例5:把用户U4修改学生学号的权限收回
revoke update(Sno) on table Student from U4;

例6:把用户U5对SC表的insert权限收回
revoke insert on table SC from U5; // 同时也会收回U5赋予其他用户的insert权限

// 角色的使用,一个角色包含的权限包括直接授予这个角色的全部权限和其他角色授予这个角色的全部权限
create role R1;
grant select, update, insert on table Student to R1;

// 将R1角色授予用户U1和角色R2,并允许他们转授相应的权限
grant R1 to U1, R2 with admin option;
revoke R1 from U1;
```

数据库完整性

- **实体完整性**：对关系模式主属性施加的完整性控制，例如Student(Sno, Sname, Sex, Byear)中Sno**不允许为空且取值唯一**，同理Course对应的Cno和SC对应的(Sno, Cno)也是如此
- **参照完整性**：对外码施加的完整性控制 (能否空值，删除和修改时的一系列问题)，维护策略如下表所示：

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/级联删除/设置为空值
修改主码值	→	可能破坏参照完整性	拒绝/级联修改/设置为空值

- **用户定义完整性**：主要参考实验
- **断言和触发器**

```
例:限制每一门课程最多60名学生选修
create assertion ASSE_SC_DB_NUM
check (60 >= (select count(*) from SC group by Cno))

例:当对SC插入元组和修改Grade值时,若为'001'号课程,Grade > 50 一律改成50分
create trigger UPDATE-SC before insert or update on SC
for each row when (:new.Cno='001')
as begin
if :new.Grade > 50
 then :new.Grade := 50
 endif
 end
```

关系数据库理论

- 不包含在任何码中的属性叫非主属性。
- **函数依赖**： 分为平凡函数依赖与非平凡函数依赖（这里只讨论后者），完全函数依赖与部分函数依赖
- 传递函数依赖的概念要了解一下
- **范式**：解决冗余、修改麻烦、插入操作异常、删除异常等问题
  - $1NF$：我们的讨论范围内，所有关系模式都是$1NF$
  - $2NF$：$\forall R(U,F), R \in 1NF$,且每个非主属性都完全函数依赖于候选码,则$R \in 2NF$。即，$2NF$中不允许存在这样的函数依赖$X \to Y$，其中$X$是码的真子集，$Y$是非主属性。
  - $3NF$:
    $\forall R(U,F), R \in 2NF$,且每个非主属性都不传递函数依赖于于候选码,则$R \in 3NF$。若$R$中没有非主属性，则其是$3NF$。
  - $BCNF$:
    $\forall R(U,F)$,若$R \in 1NF$,对于R的每一个$X \to Y, Y \notin X, X$必包含候选码,则$R \in BCNF$。它主要消除了主属性对不包含它的候选码的部分函数依赖和传递函数依赖，每一个决定因素都包含码。
- **学会求候选码**
  - L属性：必定属于候选码一部分
  - R属性：必不属于候选码
  - N属性：函数依赖两边均未出现，必定属于候选码一部分
  - LR属性：可能是候选码一部分
  - 求属性闭包
- **求最小函数依赖集的办法**
  - 函数依赖右侧多属性化为单属性
  - 对每一个函数依赖，去掉自身求闭包，如对于$A \to B$，去掉后基于剩下的函数依赖对A求闭包，看能否推出B，能则去掉，不能则保留。
  - 去除函数依赖左侧多余的属性
- 模式分解
  - 无损连接检查
  - 分解成3NF算法

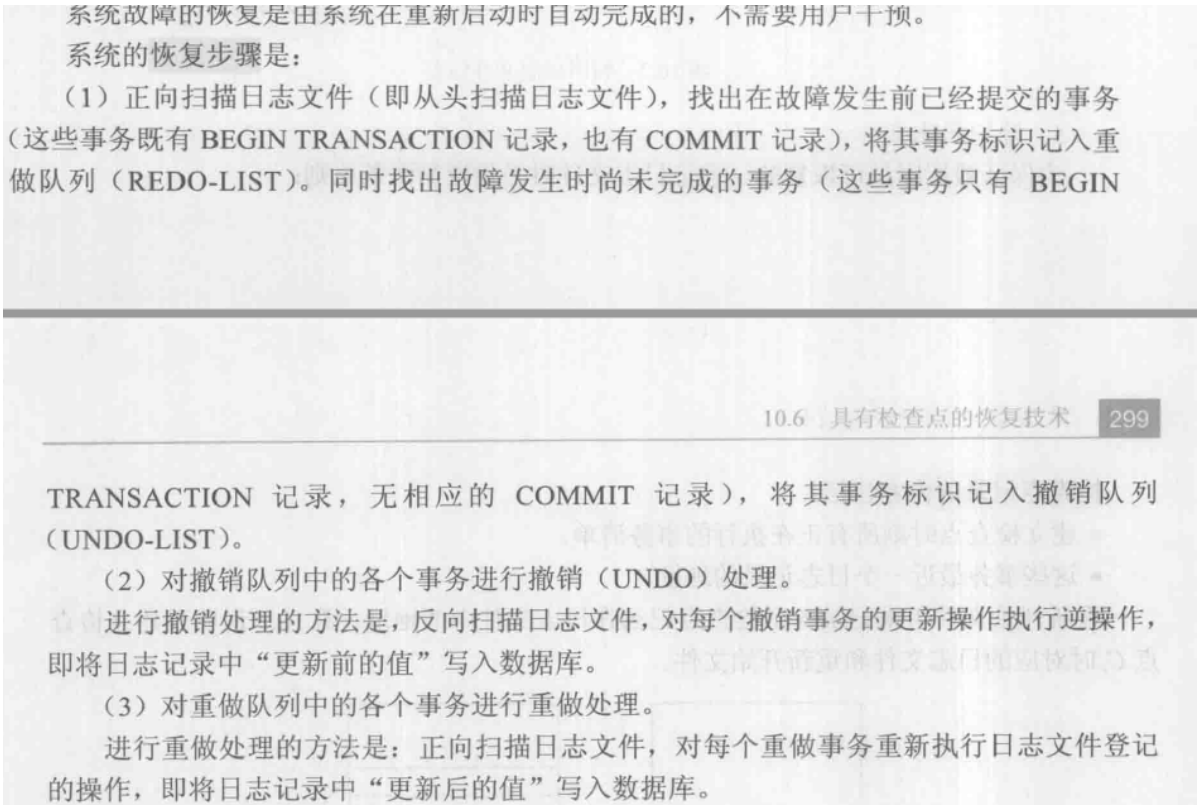
数据库恢复技术

1.故障类型

- 事务故障
- **系统故障**
- 介质故障

2.故障恢复方法

- 系统故障恢复方法



系统故障的恢复是由系统在重新启动时自动完成的，不需要用户干预。

系统的恢复步骤是：

（1）正向扫描日志文件（即从头扫描日志文件），找出在故障发生前已经提交的事务（这些事务既有 BEGIN TRANSACTION 记录，也有 COMMIT 记录），将其事务标识记入重做队列（REDO-LIST）。同时找出故障发生时尚未完成的事务（这些事务只有 BEGIN TRANSACTION 记录，无相应的 COMMIT 记录），将其事务标识记入撤销队列（UNDO-LIST）。

（2）对撤销队列中的各个事务进行撤销（UNDO）处理。

进行撤销处理的方法是，反向扫描日志文件，对每个撤销事务的更新操作执行逆操作，即将日志记录中“更新前的值”写入数据库。

（3）对重做队列中的各个事务进行重做处理。

进行重做处理的方法是：正向扫描日志文件，对每个重做事务重新执行日志文件登记的操作，即将日志记录中“更新后的值”写入数据库。

10.6 具有检查点的恢复技术 299

- 检查点恢复技术

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，如图 10.4 所示。

T_1：在检查点之前提交。

T_2：在检查点之前开始执行，在检查点之后故障点之前提交。

T_3：在检查点之前开始执行，在故障点时还未完成。

T_4：在检查点之后开始执行，在故障点之前提交。

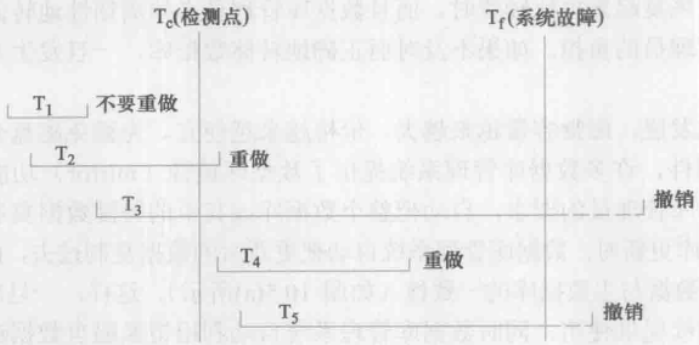


图 10.4 恢复子系统采取的不同策略

T_5：在检查点之后开始执行，在故障点时还未完成。

T_3 和 T_5 在故障发生时还未完成，所以予以撤销；T_2 和 T_4 在检查点之后才提交，它们对数据库所做的修改在故障发生时可能还在缓冲区中，尚未写入数据库，所以要重做；T_1 在检查点之前已提交，所以不必执行重做操作。

系统使用检查点方法进行恢复的步骤是：

（1）从重新开始文件中找到最后一个检查点记录在日志文件中的地址，由该地址在日志文件中找到最后一个检查点记录。

（2）由该检查点记录得到检查点建立时刻所有正在执行的事务清单 ACTIVE-LIST。

这里建立两个事务队列：

- UNDO-LIST: 需要执行 UNDO 操作的事务集合；
- REDO-LIST: 需要执行 REDO 操作的事务集合。

把 ACTIVE-LIST 暂时放入 UNDO-LIST 队列，REDO 队列暂为空。

（3）从检查点开始正向扫描日志文件。

① 如有新开始的事务 T_i，把 T_i 暂时放入 UNDO-LIST 队列；

② 如有提交的事务 T_j，把 T_j 从 UNDO-LIST 队列移到 REDO-LIST 队列；直到日志文件结束。

（4）对 UNDO-LIST 中的每个事务执行 UNDO 操作，对 REDO-LIST 中的每个事务执行 REDO 操作。

总之，检查点之前提交的不需要重做，检查点之后但故障之前提交的需要重做，故障之前未提交的要撤销

并发执行

- 三级封锁协议

表 11.1　不同级别的封锁协议和一致性保证

	X 锁		S 锁		一致性保证		
	操作结束释放	事务结束释放	操作结束释放	事务结束释放	不丢失修改	不读“脏”数据	可重复读
一级封锁协议		√			√		
二级封锁协议		√	√		√	√	
三级封锁协议		√		√	√	√	√

- 死锁：$T1$ 封锁了 $R1$，$T2$ 封锁了 $R2$，$T1$ 又请求封锁 $R2$，$T2$ 又申请封锁 $R1$。两个事务相互等待，永远不能结束
- 可串行化调度
- 两段锁协议：申请锁只能在一个阶段内完成，释放锁只能在另一个阶段内完成

---

sql语言

```sql
//建立SPJ表：
create table SPJ(
   SNO char(3),
   PNO char(3),
   JNO char(3),
   QTY int,
    primary key(SNO,PNO,JNO),
    foreign key(SNO) references S(SNO)
    略
);

//添加外码约束
alter table SPJ add constraint FK_SPJ_PNO foreign key(PNO) references P(PNO)

//删除列
alter table SC drop Grade

//修改字段
alter table SC modify SNO char(12)

//建立索引
create unique index Stusno on Student(SNO)

//查询没有使用天津供应商生产的红色零件的工程号JNO(exists语句里外两个表相同时应该取别名)
select JNO from SPJ S1 where not exists(
    select * from SPJ S2 where S1.JNO=S2.JNO and PNO in (select PNO from P where COLOR='红')
    and SNO in (select SNO from S where CITY='天津')
)

insert into SPJ values(略)

```

```
//为三建工程项目建立一个供应情况的视图
create view V_SPJ as select SNO,PNO,QTY from SPJ
where JNO in (select JNO from J where JNAME='三建工程')
```