2018 年计算机学院研究生《分布式数据库》试题（共 60 分）

1.(20 分) 设计，实现，关系代数，SQL 语句

某公司由总部和三个分公司组成，分别位于北京（含总部，编号 A0），上海（编号 A1）和广州（编号 A2），现对办公设备进行资产管理，涉及到的全局关系模式如下：

Branches(Did,Dname,Address,Mgrno)，解释：分公司基本信息表，其中：Did 为分公司编号，Dname 为分公司名称，Address 是分公司地址，Mgrno 是分公司负责人编号；

Employer(Eid,Ename,Etitle,Did)，解释：员工基本信息表，其中：Eid 为员工编号，Ename 为员工姓名，Etitle 为员工职位，Did 是员工所属的分公司编号；

Devices(Did,Dname,Dtype,Price,Pdate,Uid)，解释：设备基本信息表，其中：Did 是设备编号，Dname 是设备名称，Dtype 是设备类型，Price 是设备价格，Pdate 是设备购买时间，Uid 是设备使用人员工编号；

数据管理的基本需求是：(1) 所有分公司的基本信息都保存在总部场地；(2) 各分公司保存各自的员工信息，总部场地保存全部员工信息；(3) 单价≤2 万元的设备信息保存在使用者场地，其他设备信息保存在总部场地。

1) 按要求进行分片设计，要求写出分片定义，画出分片图，并指明分片类型；

2) 按需求完成分配设计，指出分配类型；

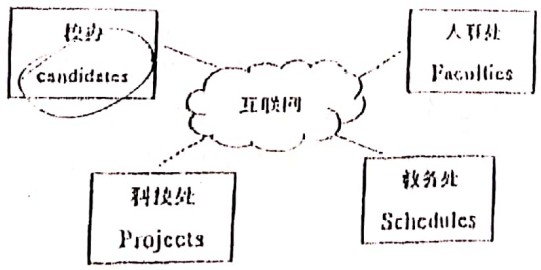
3) 查询上海分公司员工张龙所使用的于 2018 年之后购买的全部设备的基本信息，要求查询结果包括设备编号、设备名称、设备价格、购买时间，请写出查询的关系代数形式和 SQL 语句形式，并画出全局查询图；

4) 对 3)题的查询进行全局优化，要求写出中间过程；

5) 在全局优化的基础上进行片段优化，要求写出中间过程。

2.(10 分)

学校拟选择部分教师与企业进行合作科研，共有 100 名教师报名，提供了教师信号（Fid），保存在 Candidates 表中，为择优录取，校办要查询这 100 名教师的基本信息（Faculties 表，位于人事处场地）、科研项目信息（Projects 表，位于科技处场地）、教学信息（Schedules，位于教务处场地），各场地通过互联网连接，数据分布及关系模型图如下：



教师基本信息表 Faculties（单位：字节）

Card(Faculties)=3000 人，Length(Employees)=300

	FID	Fname	Fage	…
length	8	20	1	…
val	3000	2800	40	…
说明	教师编号	教师姓名	年龄	…

科研项目表 Projects（单位：字节）

Card(Projects)=30000 项，Length(Projects)=200

	PID	FID	Pname	Sdate	…
length	15	8	100	8	…
val	30000	2500	30000	25000	…
说明	项目编号	负责人教师编号	负责人教师姓名	立项时间	…

教学情况表 Schedules（单位：字节）

Card(Schedules)=120000 门次，Length(Schedules)=150

	CID	Cname	Semester	FID	…
length	15	60	20	8	…
val	4000	4000	60	2500	…
说明	课程编号	课程名称	上课学期	主讲教师编号	

假设所有数据均满足均匀分布，问：

1) 如果采用全连接技术，执行场地在校办场地，请设计查询执行策略并计算传输代价；

2) 如果采用半连接技术，请给出优化的查询执行策略并计算传输代价。

3. (12 分)

存在如下信息，场地 1：货品信息表 Article(Ano, Aname, Price, Sold, Total)，用户信息表 User(Uid, name, Tel, Type)，场地 2：购买信息表 Purchaser(Pid, Uid, Ano, Num, Money)；场地 3：电商账户（B_account），场地 4：用户账号（U_account）。假设用户 A 在某电商的一次购物为一个分布式事务，具体如下：步骤 1：客户 A 购买了商品 X，修改该货品 X 的售出数量(Sold)；步骤 2：写用户 A 的购买信息 Y 到购买信息表(Purchaser)；步骤 3：将用户 A 购买商品 X 的花费 Z 从用户账号（U_account）转账到电商账户（B_account）中。请回答：

1） 将该分布式事务分解为相应的子事务；

2） 应用进程代理模型概述该业务流程的实现过程；

3） 如果采用集中式两段提交协议，写出提交过程；

4. (10 分)

某代售网站存在如下信息：航班信息表 Flight (Fno, Date, Price, Sold, Capacity, Co.)，客户信息表 Customer(Cid, Cname, Fno, CDate)。假设：场地 1 存储南航的航班信息（Flight_S）和其客户信息（Customer_S），还有代收银行账户(B_account)；场地 2 存储国航的航班信息（Flight_Z）和其客户信息（Customer_Z），场地 3 存储用户账号（U_account）。若用户 U 在该代售网站一次购买南航和国航混合航班的业务为一个分布式事务，具体如下：步骤 1：客户 U 购买了南航航班 S 和国航航班 Z 各 1 张机票，修改 Flight_S 和 Flight_Z 的售出数量（Sold）；步骤 2：分别写用户 U 的南航购买信息 CS 和国航购买信息 CZ 到 Customer_S 和（Customer_Z）中；步骤 3，将用户 U 购买信息 CS 和 CZ 的花费 M 从用户账号（U_account）转账到电商账户（B_account）中。

若存在另一用户 V，同时购买同 U 同伴的航班，也进行相同的操作，请回答：

1） 几个场地上存在冲突操作？若存在冲突操作，请写出冲突对；

2）针对上述两个分布式事务，请构建一全局历程 H，同时写出：a）局部可串行化、全局也可串行化的局部历程；b)局部可串行化、全局不可串行化的局部历程。

5. (8 分)

对比分析维护多副本一致性协议 Paxos、反熵、NWR 的异同，请写出：各自的优势和不足？分别适合于什么场景？

解：1）对Branches进行分片，

$B_0 = \sigma_{Bid=A0}(Branches)$

$B_1 = \sigma_{Bid=A1}(Branches)$

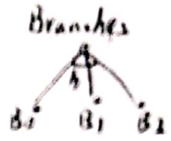
$B_2 = \sigma_{Bid=A2}(Branches)$

分片类型为水平分片

分片条件
Bid = A0
Bid = A1
Bid = A2

19

分片树



对Employee进行分片

$E_0 = \sigma_{Bid=A0}(Employee)$

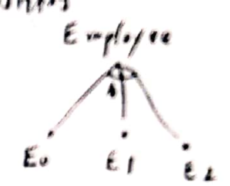
$E_1 = \sigma_{Bid=A1}(Employee)$

$E_2 = \sigma_{Bid=A2}(Employee)$

分片类型为水平分片

分片条件
Bid = A0
Bid = A1
Bid = A2

分片树



对Devices进行分片

$D_0 = Devices \propto E_0$

$D_1 = Devices \propto E_1$

$D_2 = Devices \propto E_2$

分片类型为水平分片

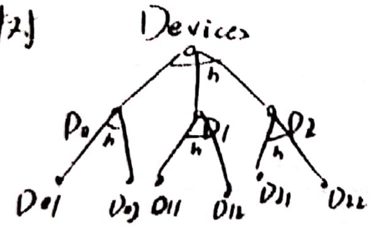
+4

$D_{01} = \sigma_{price<20000}(D_0)$, $D_{02} = \sigma_{price\geq 20000}(D_0)$

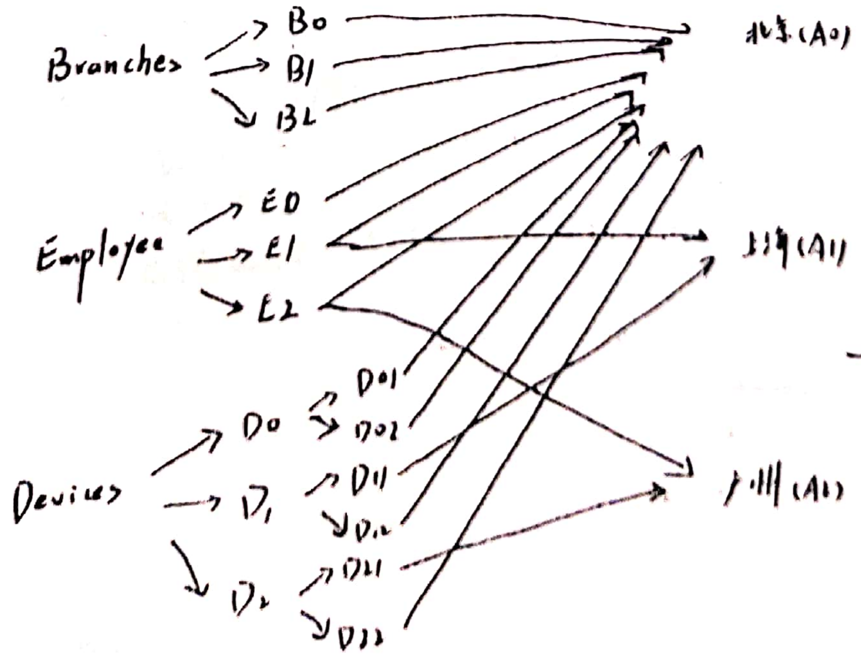
$D_{11} = \sigma_{price<20000}(D_1)$, $D_{12} = \sigma_{price\geq 20000}(D_1)$

$D_{21} = \sigma_{price<20000}(D_2)$, $D_{22} = \sigma_{price\geq 20000}(D_2)$

分片树



2）分配设计如图：



分配类型：

Branches为分割式分配（非复制分配）

Employee为部分复制分配

+4

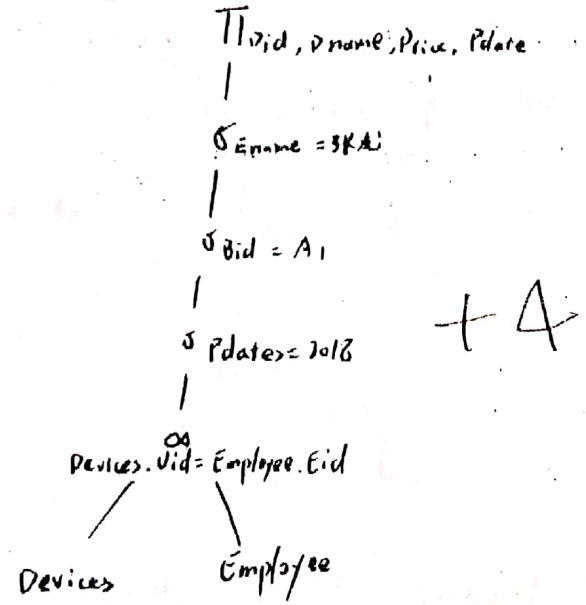
Devices为分割式分配（非复制分配）

3) $\Pi_{Did, Dname, Price, Pdate}\ \sigma_{Ename=张杰\ AND\ Bid=A_1\ AND\ Pdate>=2018}$ (Devices $\underset{Devices.Did=Employee.Eid}{\infty}$ Employee)

SQL形式：

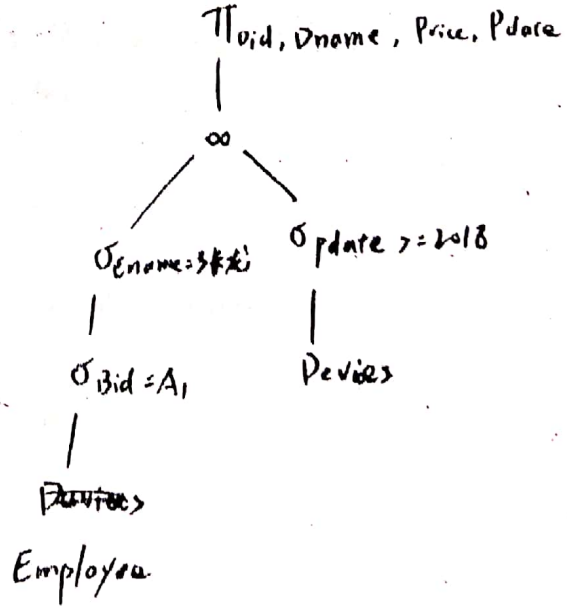
```
SELECT  Did, Dname, Price, Pdate
FROM    Employee, Devices
WHERE   ~~Employee.Eid = Devices.Did AND~~ Devices.Did=Employee.Eid AND
        Employee.Ename = 张杰 AND
        Employee.Bid = A1 AND
        Devices.Pdate >= 2018
```

查询树

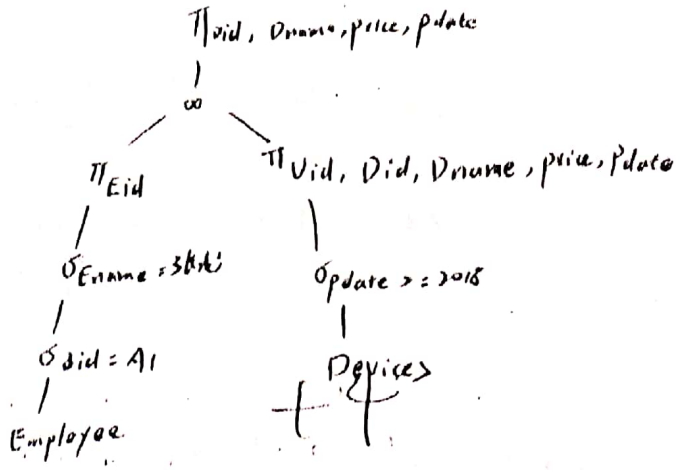


+4

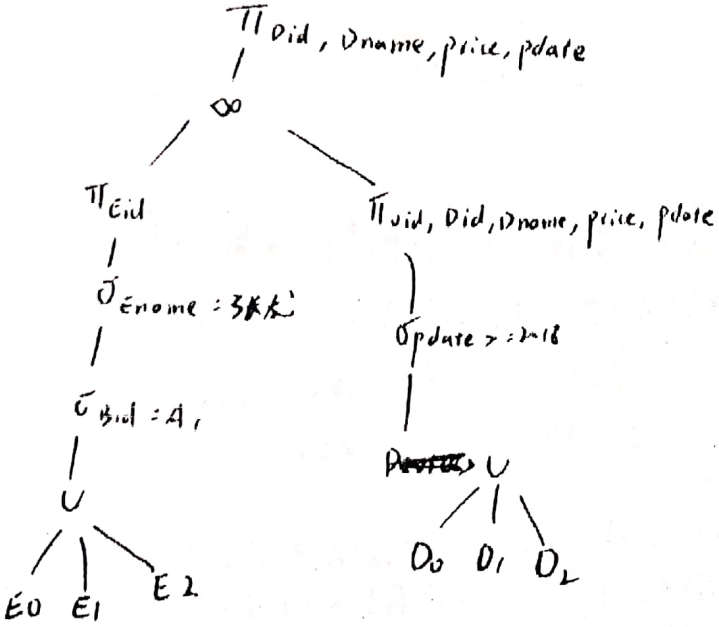
4) 对3)中查询树进行主要优化
即先进行一元运算，
如右图
先将选择运算下移

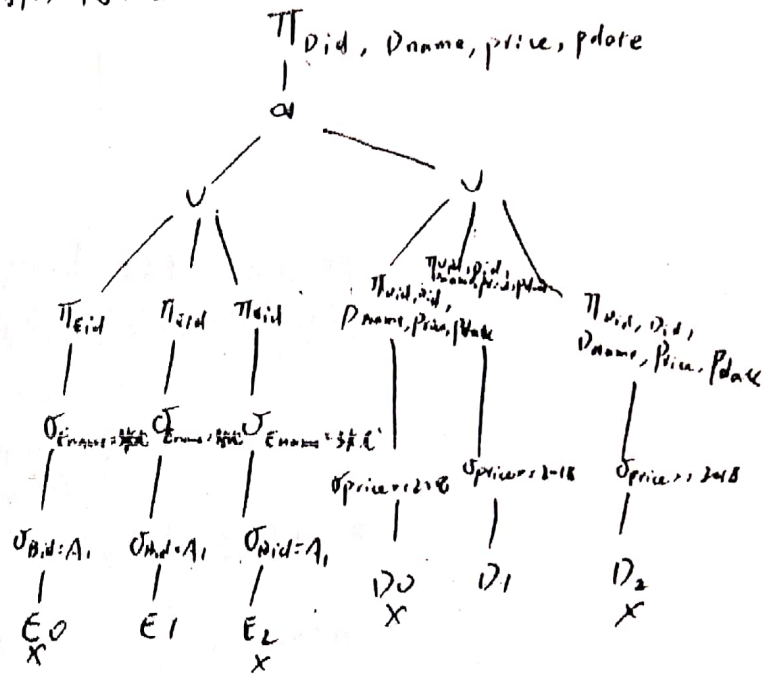


再优先进行投影运算
如右图

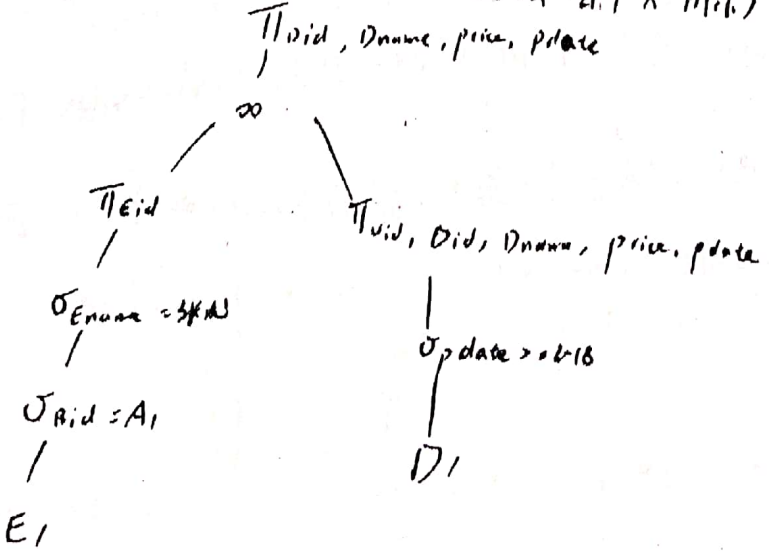


3) 进行片段优化，首先对Employee和Devices分片
回1)，得左图，再一步选择和投影下推，得右图





之后，来为限严连接条件不满足的部分(如图中X所示)
得



2、解：1）如果采用全连接技术，由于执行场地在校办场地，此时应将人事处场地上的表 Faculties表、科技处场地上的 Projects表，教务处场地上的 Schedules表中属性传送到校办场地，然后进行全连接

有：

Cost = Card(Faculties) × Length(Employees) + Card(Projects) × Length(Projects)
+ Card(Schedules) × Length(Schedules)

= 3000×500 + 30000×200 + 120000×150

= 24900000 字节

= ~~24900KB~~

=24900KB

+5

2）如果采用半连接技术，根据半连接原理，设计如下策略

1）将 Candidates表中的系别编号（Fid）传到分别 Faculties表、Projects表，Schedules表中

Cost1 = 3×100×8 = 2400 字节

= 2.4KB

2）将其与Faculties表、Projects表、Schedules表作连接

根据元组的大小，作估计得

与 Faculties 作连接后所得元组约有100个（Fid为主键）

与 Projects 作连接后所得元组有 $\frac{100}{1500}$ × 30000 = 1200 个

与 Schedules 作连接后所得元组有 $\frac{100}{1500}$ × 120000 = 4800 个

3）将2）中作连接后得到的结果传回校办场地

Cost2 = 100×300 + 1200×200 + 4800×150

= 30000 + 240000 + 720000

= 990000 字节

= 990KB

+5

总传输代价 Cost = Cost1 + Cost2 = 2.4 + 990 = 992.4KB

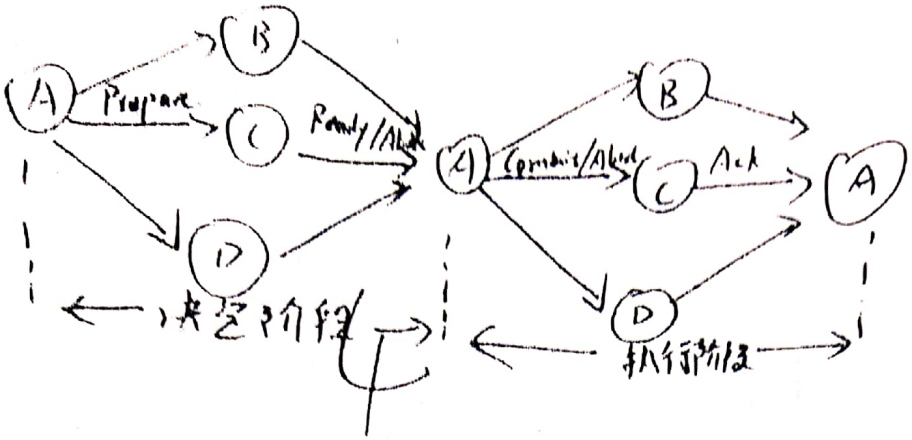
5. 解：1）当子事务指一个分布式事务在某个场地上操作的集合
（一）（二）将其分解为
子事务1（场地1）：客户A购买了商品X后修改X的售出数量（Sold）
子事务2（场地2）：写用户A的购买信息，订购购买信息表
子事务3（场地3）：将用户A购买商品A所花费的金额加到B电商账户里（B_account）
子事务4（场地4）：将用户A购买商品A所花费的金额从用户账号（U_account）中减掉

2）根据进程模型，有
根代理 ROOT

```
begin_transaction
    input (amount, U_account, B_account, buy_Ano, buy_number, from_account,
                                                              to_account)
EXEC SQL SELECT amount
         INTO fromamount
         FROM Account
SXEC SQL SELECT Sold INTO number1,
         INTO number buy_number INTO number2
         FROM Article, Purchase
         WHERE Article.Ano = Purchase.Ano buy.Ano
  If not number1 <= number2
     Output ("Error");
     Abort;
ELSE ELSE
     Create AGENT1;
     Send (agent1, buy_Ano, buy_number);
     Receive (agent1, flag, name);
     If flag == FALSE
        output ("Error");
        Abort;
  ELSE.
  ELSE Create Ag AGENT2;
  Receive (agent2, flag, name)
  If flag = FALSE
     Output ("failure");
     Abort;
```

```
Ø ELSE   EXEC SQL SELECT Xamount
         INTO temp
         FROM B_acount
         WHERE Accouns = B_account
               from_accont
  If   tem < transfer_amount then
         Output ("failure");
         Abort;
  ElSE   EXEC SQL UPDATE Acount
         SET Amount = Amount - transfer_amount,
         WHERE Accouns = fromacc; B_account;
                   from_account
       Create AGENT3;
       SEo
       send (agent3, transfer_amount, to_account)
       Commit;

  End

子代理1 (AGENT1):
   Receive (ROOT, buy_Ano, buy_number)
   EXEC SET SQL UPDATE Sold
        SET  Sold = Sold - buy_number;
        WHERE Article.Ano == buy_Ano

子代理2 (AGENT2)
   Receive (ROOT, buy_Ano, buy_number)
   EXEC SQL UPDATE hNum
   SET  Num = Num + buy_number
   WHAERE Purchaser_Anm == buy_Ano;
                    Ano

子代理3 (AGENT3)

   Receive (ROOT, Xtransfer_amount, to_account)
   EXEC SQL UPDATE ACCOUNT
        SET aAmount = amount + transfer_amount
        WHERE U_account = to_acc;
```

3) 设根AGENT ROOT为协调者，子代理AGENT 1、2、3为参与者，分别用A、B、C、D表示，则有



9

场地1上有

W(Sold) ~~W(Customer_S)~~ R(Customer_S)

场地2上有

~~W(Sold) R~~

~~存场地3上有~~

用于保存数量在航班信息表中

设 Flight_S 为 x　　Customer_S 为 y

Flight_Z 为 u　　Customer_Z 为 v

B_account 为 a　　U_account 为 b

别场地上有

r(x) w(x) r(y) w(y) w(b a)

场地2上有

r(u) w(u) r(v) w(v)

场地3上有

w(b)

设 Flight_S 为 x

Customer_S 为 y

Flight_Z 为 u

Customer_Z 为 v

B_account 为 a

U_account 为 b

场地1、2、3上都存在冲突

冲突对

场地1：

W(U_Sold) 与 W(V_sold)

W(U_Cs) 与 W(V_Cs)

~~W(~~

场地2：

W(U_sold) 与 W(V_sold)

W(V_CB2) 与 W(V_Cz)

场地3：W(U_transfer) 与 W(V_transfer)

c) 全局历程

a) 场地1

T_u	T_v
r(x)	r(x)
w(x)	⋮
r(y)	⋮
w(y)	⋮
r(a)	⋮
w(a)	⋮
	w(x)
	r(y)
	w(y)
	r(a)
	w(a)

场地2

T_u	T_v
r(u)	r(u)
w(u)	⋮
r(v)	⋮
w(v)	⋮
	w(u)
	r(v)
	w(v)

$T_u < T_v$

场地3

T_u	T_v
r(b)	r(b)
w(b)	⋮
	w(b)

b) 场地1

T_u	T_v
r(x)	r(x)
w(x)	⋮
r(y)	⋮
w(y)	⋮
r(a)	⋮
w(a)	⋮
	w(x)
	r(y)
	w(y)
	r(a)
	w(a)

$T_u < T_v$

场地2

T_u	T_v
r(u)	r(u)
⋮	w(u)
⋮	r(v)
⋮	w(v)
w(u)	
r(v)	
w(v)	

$T_u > T_v$

场地3

T_u	T_v
r(b)	r(b)
w(b)	⋮
	w(b)

1)

Paxos:

Paxos用于实现分布式系统的多个节点数据一致性的协议，通过消息传递的方式，在多个冲突请求中选出一个从而达到分布式系统中各节点数据一致

是一个民主选举的过程

允许消息丢失或重复，但不会出现损坏

反熵：消息被传递直到它被新的消息取代。

反熵协议可用于在一组参与者中共享可靠消息。运用协调机制

只有临主节点才能修改

系统中的两个节点存在三种通信方式

push：A将数据推送给B，B更新A中比自己新的数据

pull：A将摘要数据推给B，B根据其来选择版本号比A新的数据给A，A更新

push-pull，与pull类似，多了一步，A→B，B更新。

NWR：N：复制节点数

R：成功读操作的最小节点数

W：成功写操作的最小节点数

如果系统要求满足强一致性，配置要求W+R>N

满足上述条件使得客户端每次读取，都至少读到一个最新版本，从而不会读到旧数据