一、 何为分布式数据库系统？一个分布式数据库系统有哪些特点？

答案：分布式数据库系统通俗地说，是物理上分散而逻辑上集中的数据库系统。分布式数据库系统使用计算机网络将地理位置分散而管理和控制又需要不同程度集中的多个逻辑单位连接起来，共同组成一个统一的数据库系统。因此，分布式数据库系统可以看成是计算机网络与数据库系统的有机结合。 一个分布式数据库系统具有如下特点：

物理分布性，即分布式数据库系统中的数据不是存储在一个站点上，而是分散存储在由计算机网络连接起来的多个站点上，而且这种分散存储对用户来说是感觉不到的。

逻辑整体性，分布式数据库系统中的数据物理上是分散在各个站点中，但这些分散的数据逻辑上却构成一个整体，它们被分布式数据库系统的所有用户共享，并由一个分布式数据库管理系统统一管理，它使得“分布”对用户来说是透明的。

站点自治性，也称为场地自治性，各站点上的数据由本地的 DBMS 管理，具有自治处理能力，完成本站点的应用，这是分布式数据库系统与多处理机系统的区别。另外，由以上三个分布式数据库系统的基本特点还可以导出它的其它特点，即：数据分布透明性、集中与自治相结合的控制机制、存在适当的数据冗余度、事务管理的分布性。

二、 简述分布式数据库的模式结构和各层模式的概念。

分布式数据库是多层的，国内分为四层：

全局外层：全局外模式，是全局应用的用户视图，所以也称全局试图。它为全局概念模式的子集，表示全局应用所涉及的数据库部分。

全局概念层：全局概念模式、分片模式和分配模式

全局概念模式描述分布式数据库中全局数据的逻辑结构和数据特性，与集中式数据库中的概念模式是集中式数据库的概念视图一样，全局概念模式是分布式数据库的全局概念视图。分片模式用于说明如何放置数据库的分片部分。分布式数据库可划分为许多逻辑片，定义片段、片段与概念模式之间的映射关系。分配模式是根据选定的数据分布策略，定义各片段的物理存放站点。

局部概念层：局部概念模式是全局概念模式的子集。 局部内层：局部内模式

局部内模式是分布式数据库中关于物理数据库的描述，类同集中式数据库中的内模式，但其描述的内容不仅包含只局部于本站点的数据的存储描述，还包括全局数据在本站点的存储描述。

三、 简述分布式数据库系统中的分布透明性，举例说明分布式数据库简单查询的各级分布透明性问题。

分布式数据库中的分布透明性即分布独立性，指用户或用户程序使用分布式数据库如同使用集中式数据库那样，不必关心全局数据的分布情况，包括全局数据的逻辑分片情况、逻辑片段的站点位置分配情况，以及各站点上数据库的数据模型等。即全局数据的逻辑分片、片段的物理位置分配，各站点数据库的数据模型等情况对用户和用户程序透明。

分布透明性包括三个层次：分片透明性，是分布透明性中的最高层；位置透明性，也称分配透明性，是分布透明性的中间层；局部数据模型透明性，也称局部映像透明性，即与各站点上数据库的数据模型无关，是分布透明性的最底层。

四、 讨论分布式数据库更新应用中的各级分布透明性问题。

分片透明性：应用程序如同数据库不是分布的那样来执行更新操作，编程人员不必知道被更新的属性是否是分片模式的定义中使用的属性。
位置透明性：应用程序员必须要知道分片情形，并将给出明确的处理。这是一种非常简单的更新应用，最后四个语句可以任何次序或并行执行。

本地映像透明性：应用程序员必须明确地处理片段的位置。若为更新应用，还必须考虑片段的复制问题

五、 数据库设计中分片设计的基本目的是什么？何为水平分片？举例说明初级分片和导出分片的方法。

分片设计的基本目的是产生一个对全局数据合适的划分方案。使用这种方案得到的片段作为分布式数据库中数据的分配和存储单位时，不但能够减少应用中的操作量，而且能够对于应用具有最大可能的本地性，即使得各片段位于其使用最多的站点，或者说，使绝大多数应用所使用的数据位于该应用的原发站点。但是，不是所有的全局数据都必须进行分片，应考虑到有可能一个全局关系根本不需要分片。特别是，如果分片一个全局关系所能够获得的好处太小，不足以补偿因分片造成的开销，就不必须对该全局关系进行分片。
水平分片是对全局关系执行“选择”操作，把具有相同性质的元组进行分组，构成若干个不相交的子集。水平分片的方法可归为初级分片和导出分片两类。
初级分片：以关系自身的属性性质为基础，执行“选择”操作，将该关系分片成若干个不相交的片段。
例如：S(S#,SNAME,AGE,SEX)

Define fragment S1 as select * from s where sex = ‘M’
Define fragment S2 as select * from s where sex = ‘F’

导出分片：全局关系的导出式水平分片不是以其自身的属性性质为基础，而是从另一个关系的属性性质或水平片段推导出来的。采用导出分片可使片段与片段之间的“连接”变得更容易。
例如：设全局关系 SC(S#,C#,GRADE)

S(S#,SNAME,AGE,SEX)

若要将 SC 划分为男生的各门课成绩和女生的各门课成绩。这就不可能从 SC 本身的属性性质来执行选择，必须从关系 S 的属性性质或水平片段来导出。

```
define fragment SC 1 as
select SC．S#，C#，GRADE from SC，S
where SC．S#=S.S# and SEX=’M’
define fragment SC2 as
select SC.S#，C#，GRADE from SC，S
where SC.S#=S.S# and  SEX=’F’
```

如果 S 已经进行水平分片，分为 SF、和 SM，分别为男生全体和女生全体，则上述的片段定义可以基于片段 SF 和 SM 导出：

define fragment SC 1 as

select*from SC where S# in(select SF.S# from SF)

define fragment SC2 as

Select * from SC where S# in(select SM.S# from SM)

六、 水平分片正确性原则的三个条件是什么？请说明它们的意义。

1)完整性条件。各片段定义中的限定语集合必须是完整的，即至少是它们允许值的集合。例如：SEX={’M’,’F’} 季节={春，夏，秋，冬}

2)可重构条件。如果限定语集合是完整的，则通过并操作总能重构全局关系。 3)不相交条件。如果限定语之间是互斥的，它们的片段必不相交。 其意义是确定一组合适的不相交的，完整的限定语。

七、 数据库的片段位置分配设计中，何为冗余分配？请简述其两种设计方法。

在确定数据片段的位置分配时，冗余分配即要每个片段映射到一个或多个站点上。冗余分配的设计较为复杂，使用冗余分配，设计者必须决定每一片段复制的程度。复制的利益随着检索与更新间的比值而增加，因为数据维护的一致性需要将更新传播到所有副本。然而，系统可以允许临时不一致性，在这种情况下，复制变得更加有用。此外，复制增加了从故障中恢复的能力，这是因为同一数据的几个副本不大可能同时全部丢失或破坏，而且当某一故障损坏被经常访问的一些副本时，应用可以访问其他的副本。 冗余分配的两种设计方法：

1)“所有得益站点”法：首先确定非复制问题的解，然后在全部站点中确定一组站点，给这组中的每一站点分配片段的一个副本，这样做所得到的好处要比为此而付出的费用合算。

2)“附加复制”法：首先确定非复制的问题的解，然后从最有益处起逐步附加复制的副本，此过程直到“附加复制”已无明显好处时结束。这种方法是典型的启迪式方法。采用这种方法考虑到随着冗余度的增加得益逐渐减少。一般，当一个片段只有两三个副本时，系统的得益在增加；但当副本数再增加时，系统的得益就不再明显增加。

八、 举例说明数据片段分配的费用和得益估算方法。

为了进行数据片段分配的费用和得益估算，假定：

i 表示片段的下标

j 表示站点的下标

k 表示应用的下标

F_{kj}表示应用 k 在站点 j 上被激活的频率

R_{ki}表示应用 k 被激活一次，对片段 i 进行检索访问的次数

U_{ki}表示应用 k 被激活一次，对片段 i 进行更新访问的次数

$N_{ki} = R_{ki} + U_{ki}$应用 k 被激活一次，访问片段 i 的总次数

(1) 水平分片情况

1) 非冗余分配使用"最佳适应"方法。即将片段 R_i 分配到访问 R_i 次数最多的那个站点上。在站点 j 上 R_i 的本地访问次数是：

$$B_{ij} = \sum_k F_{kj} * N_{ki}$$

估算：$\max(B_{ij}) = B_{ij'}$，片段 R_i 就分配在站点 j' 上。

2) 冗余分配使用"所有得益站点"方法。即将片段 R_i 的副本分配到所有得益站点 j 上。所谓所有得益站点是指在这些站点上，应用的检索访问费用总比从任何一个其他站点发出的应用对 R_i 进行更新访问的费用要低。估算这个差额：

$$B_{ij} = \sum_k F_{kj} * R_{ki} - c * \sum_k \sum_{j' \neq j} F_{kj'} * U_{ki}$$

其中：c 为度量更新访问费用与检索访问费用之比的一个常数，$c \geqslant 0$。

如果：$B_{ij} > 0$，则站点 j 为得益站点，将存放片段 R_i 的一个副本。

3) 冗余分配使用"附加复制"方法。令 D_i 表示片段 R_i 冗余度(副本的个数)；F_i 表示 R_i 在每个站点全都复制的得益。D_i 与 F_i 之间存在如下关系：

$$\beta(D_i) = (1 - 2 * * (1 - D_i)) * F_i$$

注意：$\beta(1) = 0$，$\beta(2) = F_i/2$，$\beta(3) = 3F_i/4$ 等。

修改(2)中的公式得下面求站点 j 上引入 R_i 新副本的得益公式：

$$B_{ij} = \sum_k F_{kj} * R_{ki} - c * \sum_k \sum_{j' \neq j} F_{kj'} * U_{ki} + \beta(D_i)$$

(2) 垂直分片情况

假定把站点 r 上的关系 R 垂直分片成两个片段 R_s 和 R_t，并将 R_s 和 R_t 分别分配在站点 s 和站点 t 上，然后将应用分组并估算它们的得益情况，见图 2.4。

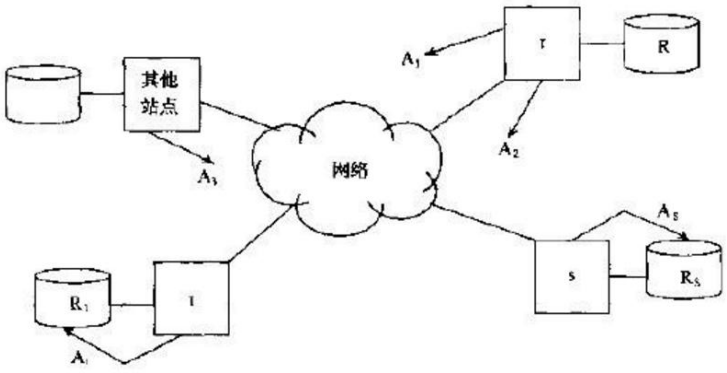


图 2.4 垂直分片的得益估算

1）应用组 A_s：自站点 s 发出，它们只使用 R_s，因而是本地应用，得益：

$$BA_s = \sum F_{ks} * N_{ki}(k \in A_s)$$

2）应用组 A_t：自站点 t 发出，它们只使用 R_t，因而是本地应用，得益：

$$BA_t = \sum F_{kt} * N_{ki}(k \in A_t)$$

3）应用组 A_1：由站点 r 发出，原先使用 R_r 或 R_s，现在这些应用需要进行一次额外的远程访问。损失：

$$BA_1 = \sum F_{kr} * N_{ki}(k \in A_1)$$

4）应用组 A_2：由站点 r 发出，原先使用 R（本地访问），而现在这些应用需要进行两次额外的远程访问。损失：

$$BA_2 = \sum F_{kr} * N_{ki}(k \in A_2)$$

5）应用组 A_3：位于不同于 r、s 或 t 的站点上，它们要访问 R_r 和 R_s 这两者的属性；现在这些应用需要一次额外的远程访问。损失：

$$BA_3 = \sum\sum F_{kj} * N_{ki}(k \in A_3, j \neq r,s,t)$$

这种分片和分配的得益为

$$B_{rst} = BA_s + BA_t - BA_1 - BA_2 - BA_3$$

九、 请分析分布式查询策略优化的重要性（参照例 3.1 举例说明）。

例 3.1 在教学数据库中，有

S（S#，SNAME，AGE，SEX）有 10 ** 4 个元组，在站点 A 存放

C（C#，CNAME，TEACHER）有 10 ** 5 个元组，在站点 B 存放

SC（S#，C#，GRADE）有 10 ** 6 个元组，在站点 A 存放

假定：若每个元组的长度均为 100 b

通信系统的传输速度为 10 ** 4 b/s

通信延迟时间为 1s

问题：要求查出所有选修'MATHS'课的男学生的学号和姓名。

解：在分片透明性的 DDBMS 支持下，SQL 语句是：

```
SELECT S#, SNAME FROM S,SC,C
    WHERE S.S# =SC.S# AND SC.C# =C.C# (连接条件)
      AND SEX= 'M' AND CNAME= 'MATHS' (选择条件)
```

通信代价的估算公式是：

```
T = 传输延迟时间 C0 + (传输的数据量 X * 数据传输速率 C1)
  = (传输次数 * 1) + (传输的 bit 数 / 10 ** 4)
```

为了实现这一查询，可以有六种可能的查询策略，如下所示：

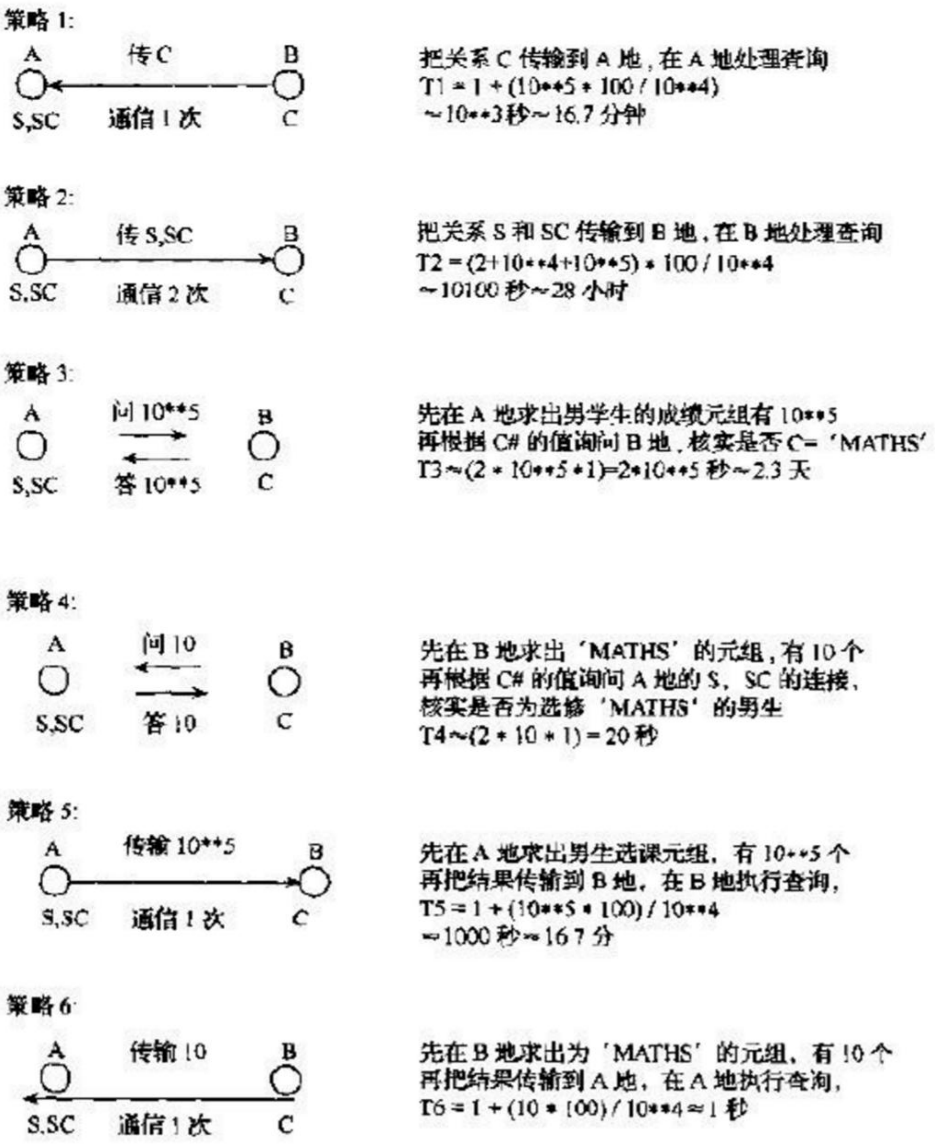


图 3.1　分布式数据库查询策略的比较

估算结果列如下表所示

表 3.1　不同查询策略结果的比较

处理策略	方 法	时　间
1	把 C 送到 A 地，在 A 地处理查询	16.7 秒
2	把 S,SC 送到 B 地，在 B 地处理查询	28 小时
3	对每个男生的成绩核查相应的课程名	2.3 天
4	取课程名为'MATHS'的课程号，核查为男生的记录	20 秒
5	把男生记录送到 B 地，在 B 地处理查询	16.7 分
6	把'MATHS'的记录送到 A 地，在 A 地处理查询	1 秒

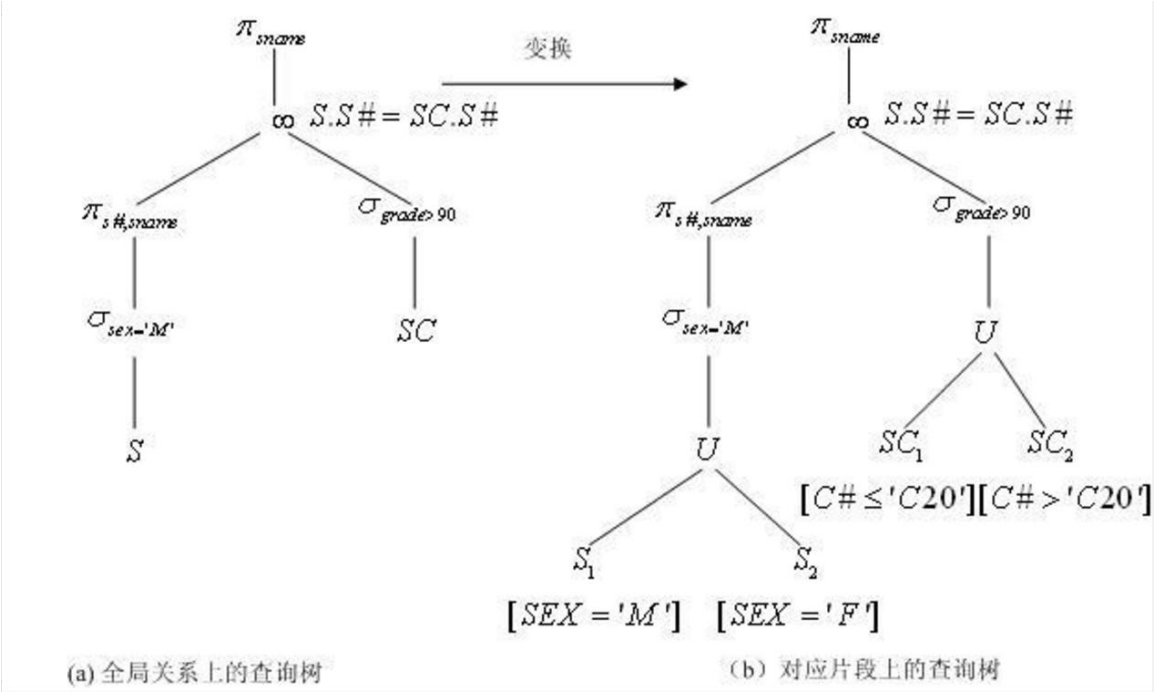
由此可见，一个好的查询处理应该使数据的传输量和通信次数最少，这样才能使查询所花费的数据传输和/或通信时间减少，从而减少查询的总代价。

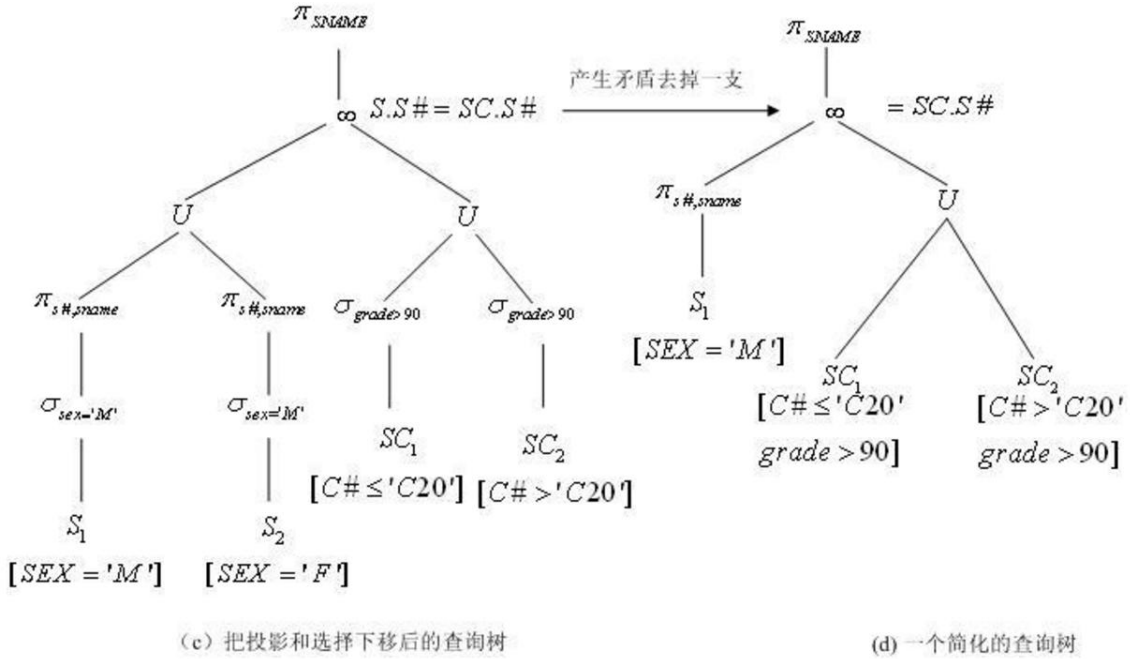
十、考虑教学数据库中的全局关系 S（s#，sname，age，sex）和 SC（s#，c#，grade）被水平分片。S 的分片限定语为：sex=“M”和 sex=“F”，SC 的分片限定语为：c#<=20 和 c#>20。若有查询问题：“查找至少有一门课程的成绩在 90 分以上的男学生姓名”，它的关系代数表达式为：

π_{sname}（$\sigma_{sex=”M”\wedge grade>90}$（$\sigma_{S.s\#=SC.s\#}$（S×SC））

请给出它的查询树。按等价变换准则进行变换，并给出变换的查询树。

它的查询树如下图 a，按等价变换准则进行变换后，得出 b, c, d 图。





十一、简述基于半连接算法的查询优化原理，举例讨论。

如果不采用半连接，而是直接把 R 送到站点 2 上执行连接操作(这里假定关系 R 的数据量小于关系 S 的数据量)的代价为

$$T_{全} = C_0 + C_1 * size(B) * card(R)$$

虽然半连接操作不具有对称性，且对于复杂连接查询会有多个半连接，各个半连接方案的代价不同，但其中总有一个方案是最优的，选作采用半连接的代价，记为 $T_{半}$。

由上述采用半连接方法表示连接操作的操作过程可知：采用半连接实现连接操作需要两次数据传输：连接属性投影结果和半连接结果。但在通常情况下，这两次数据传输的总量要远远小于传输一个整个关系的数据量，因此，一般地有 $T_{半} \ll T_{全}$。

采用半连接操作的得益与损失为

得益：当 card(R) ≫ card(R')时，可减少站点间的数据传输量。

损失：传输 $\pi_B(S) = C_0 + C_1 * (size(B) * val(B[S]))$

由此可见，采用半连接操作的分布式查询处理的言下之意是在从一个站点传送关系到另一个站点做连接之前，先除去那些与连接无关的数据，减少做连接操作的关系中的数据量，从而减少传输的代价。因此，基于半连接算法优化连接查询，其基本原理是经半连接操作，可减少操作关系的数据量，从而减少站点间数据的传输量。所以，如果只需要一个关系中的一小部分元组参与和另一个关系连接的话，这是一个使数据传输量最小化的非常有效的方案。此时有 $T_{半} < T_{全}$，采用半连接方案是合适的。在分布式数据库的查询优化中，这是常用的方法之一。

答案二：　基本原理

1. 通常有两次传输
2. 但是传输的数据量和传输整个关系相比，要远远少
3. 一般有：T 半<<T 全
4. 半连接的得益：当 card（R）>>card（R’），可减少站点间的数据传输量
5. 半连接的损失：传输 B (S) =C0+C1* size (B)* val(B[S])
6. 基本原理是在传到另一个站点做连接前，消除与连接无关的数据，减少做连接操作的数据量，从而减小传输代价

十二、假定站点 1 上的关系 R 和站点 2 上的关系 S 在属性 R.A 和 S.B 上做关于 R.A=S.B 的连接操作。请用半连接方法表示该连接操作，画出相应的示意图，给出代价估算分析。

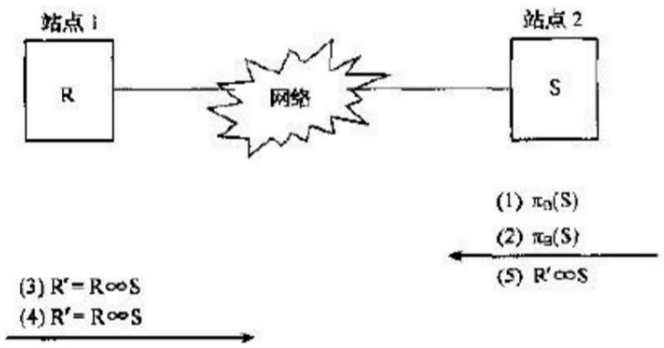
答案：当连接操作采用半连接方法表示时，有

$$R \infty_{A=B} S = (R \infty_{A=B} S) \infty_{A=B} S = (R \infty_{A=B} \pi_B(S)) \infty_{A=B} S$$

或

$$R \infty_{A=B} S = (S \infty_{A=B} R) \infty_{A=B} R = (S \infty_{A=B} \pi_A(R)) \infty_{A=B} R$$

采用半连接方法表示连接操作示意图如下图所示：



因传输代价可用下式估算：

T=C0+C1*X

则：

1）在站点 2 上作关系 S 在 R 和 S 公共属性集 B 上的投影 $\pi_B(S)$；

2）把 $\pi_B(S)$ 结果送到站点 1，代价为

$$C_0+C_1*size(B)*val(B[S])$$

size(B)为 B 属性的长度，val(B[S])为关系 S 中属性 B 上不同值的个数。

3）在站点 1 上计算半连接，设其结果为 R′，则 $R'=R\propto_{A-B}S$；

4）把 R′从站点 1 送到站点 2 的代价是：

$C_0+C_1*size(R)*card(R')$　　其中 card(R′)为 R′的元组数

5）在站点 2 上执行连接操作：　$R'\infty_{A-B}S$；

采用半连接方案的总代价为

$$T_{半R}=2C_0+C_1*(size(B)*val(B[S])+size(R')*card(R'))$$

或

$$T_{半S}=2C_0+C_1*(size(A)*val(A[R]+size(S')*card(S'))$$

十三、假定一个查询要进行关系 R_1 和 R_2 的连接，R_1 和 R_2 的数据分布如图所示。设片段大小为 F_{11}=50，F_{12}=50，F_{21}=100，F_{22}=200；数据通信代价由 C（x）=x 给出（即 C_0=0，C_1=1）；每个站点上的本地连接代价由 J（x_1，x_2）=5*（x_1+ x_2）给出，且每个站点上的并操作代价∪（x_1，x_2）=2*（x_1+ x_2）。请分别计算 FT（Q，S_1，R_1）、FT（Q，S_2，R_1）、FT（Q，S_1，R_2）、FT（Q，S_2，R_2），并据此，选择在基于直接连接算法的查询优化中哪一个关系保持分片状态。

		站　点	
		S_1	S_2
关系	R_1	F_{11}	F_{12}
	R_2	F_{21}	F_{22}

答案：当关系 R1 保持分片状态时，站点 S1 的完成时间 FR(Q,S1,R1)为：

FT（Q,S1,R1）=200+2*（100+200）+5*（50+300）=2550，其中 200 是传送 F22 的通信代价，2*（100+200）是 F21 和 F22 的并操作代价，5*（50+300）是 R2 和 F11 的连接操作代价。

同样地，FT（Q,S2,R1）=100+2*（100+200）+5*（50+300）=2450.

因此，查询的响应时间在 R1 保持分片状态时为 2550。

同样计算 R2 保持分片状态时得到

FT（Q,S1,R2）=50+2*（50+50）+5*（100+100）=1250

FT（Q,S1,R2）=50+2*（50+50）+5*（100+200）=1750

因此，在 R2 保持分片状态时，查询的响应时间 1750。由于 R1 保持分片状态时的响应时间大于 R2 保持分片状态时的响应时间，所以应选择 R2 作为保持分片状态的关系。

十四、对于如下定义的两个事务：

T_1：read（x）	T_2：read（x）
x=x+10	x=x-20
write（x）	write（x）
read（y）	read（y）
y=y-15	y=y*2
write（y）	write（y）
commit	commit

请给出各种可能的调度，并指出各自属于哪些类型的调度（串行调度、一致性调度、可串行化调度）。

答案：

对于这两个事务，可产生如下五种调度：

S_1= {$R_1(x)$, x_1=x+10, $W_1(x)$, $R_1(y)$, y_1=y−15, $W_1(y)$, C_1,
$R_2(x)$, x_2=x−20, $W_2(x)$, $R_2(y)$, y_2=y*2, $W_2(y)$, C_2}

S_2= { $R_1(x)$, x_1=x+10, $W_1(x)$, $R_2(x)$, x_2=x−20, $W_2(x)$,
$R_1(y)$, y_1=y−15, $W_1(y)$, C_1, $R_2(y)$, y_2=y*2, $W_2(y)$, C_2}

S_3= { $R_1(x)$, x_1=x+10, $W_1(x)$, $R_2(x)$, x_2=x−20, $W_2(x)$,
$R_2(y)$, y_2=y*2, $W_2(y)$, C_2, $R_1(y)$, y_1=y−15, $W_1(y)$, C_1}

S_4= { $R_2(x)$, x_2=x−20, $W_2(x)$, $R_2(y)$, y_2=y*2, $W_2(y)$, C_2,
$R_1(x)$, x_1=x+10, $W_1(x)$, $R_1(y)$, y_1=y−15, $W_1(y)$, C_1}

S_5= { $R_2(x)$, x_2=x−20, $W_2(x)$, $R_1(x)$, x_1=x+10, $W_1(x)$,
$R_2(y)$, y_2=y*2, $W_2(y)$, C_2, $R_1(y)$, y_1=y−15, $W_1(y)$, C_1}

如果将事务提交延迟到两个事务操作完成之后执行，那么就有

1）调度 S_1 和 S_4 是串行调度，也是一致性调度；

2）调度 S_2 和 S_1 的冲突操作具有相同的序，因此是等价的调度；S_2 是可串行化调度，也是一致调度；

3）调度 S_3 虽是个一致调度，但它不与 S_1 或 S_4 等价，所以 S_3 不是可串行化调度；

4）调度 S_5 和 S_4 等价，所以 S_5 是一致调度，也是可串行化调度。

由此可见：

1）同一事务集上的可串行化调度，结果未必相同；

2）一个可串行化调度必定与某个串行调度等价，且是一致调度；

3）一致调度不一定是可串行化调度；

4）同一事务集的几个可串行化调度，它们的结果未必相同。

十五、试举例说明分布式数据库在实际中的应用。