我们用**传送磁盘块数**以及**搜索磁盘次数**来度量查询计划的代价。假设磁盘系统传输一个块的平均消耗tT秒，磁盘块平均访问时间（磁盘搜索时间加上旋转延迟）为tS秒，则一次传输b个块以及S次磁盘搜索将消耗b\*tT+S\*tS秒。tT和tS的值必须针对所使用的磁盘系统进行计算（意思就是给定的，不是你算出来的），通常典型数值是tT =0.1毫秒和tS=4毫秒。**我们每次选较小的块作为外层关系，也就是nr,br作为外层关系**

*嵌套连接循环*：

最坏的情况(缓冲区只能容纳每个关系的一个数据块): 我们需要**nr\*bs+br**次块传输，需要**nr+br**次磁盘搜索次数。

最好的情况(内存有足够的空间同时容纳两个关系,此时每一数据块只需要读一次): 我们需要br+bs次块传输，加上2次磁盘搜索。

*块嵌套循环连接：*

最坏的情况: 我们需要**br\*bs+br**次块传输，2br次磁盘搜索。

最好的情况：我们需要**br+bs**次块传输，2次磁盘搜索。

*索引嵌套循环连接：*（将有B+树的那一个作为内循环，则下面的公式带入的值是外循环的tuples和blocks. 如果两个都有b+树，则选取元组树较少的作为外层关系）

块传输次数和磁盘搜索次数一样：nr\*c+br  (c = (hi+1), 也就是树的高度加1)

*并归连接：(在计算未排序好的关系是，要先计算已经排序好的再加上未排序好的)*

如果关系已排序：则块传输次数是:br+bs. 磁盘搜索次数是: ⌈br/bb⌉+⌈bs/bb⌉ （bb是分配的缓冲块，最好情况是1­­,下面的M是并归段）

如果关系未排序：先计算进行排序时需要的递归次数：⌈logm-1(br/M) ⌉。再计算进行块传输次数: br\*(2\*⌈log取下整[M/bb]-1 (br/M) ⌉+1)。磁盘的搜索次数是:

2\*⌈br/M⌉+⌈br/bb⌉\*(2\*⌈log取下整[M/bb]-1(br/M)⌉-1)

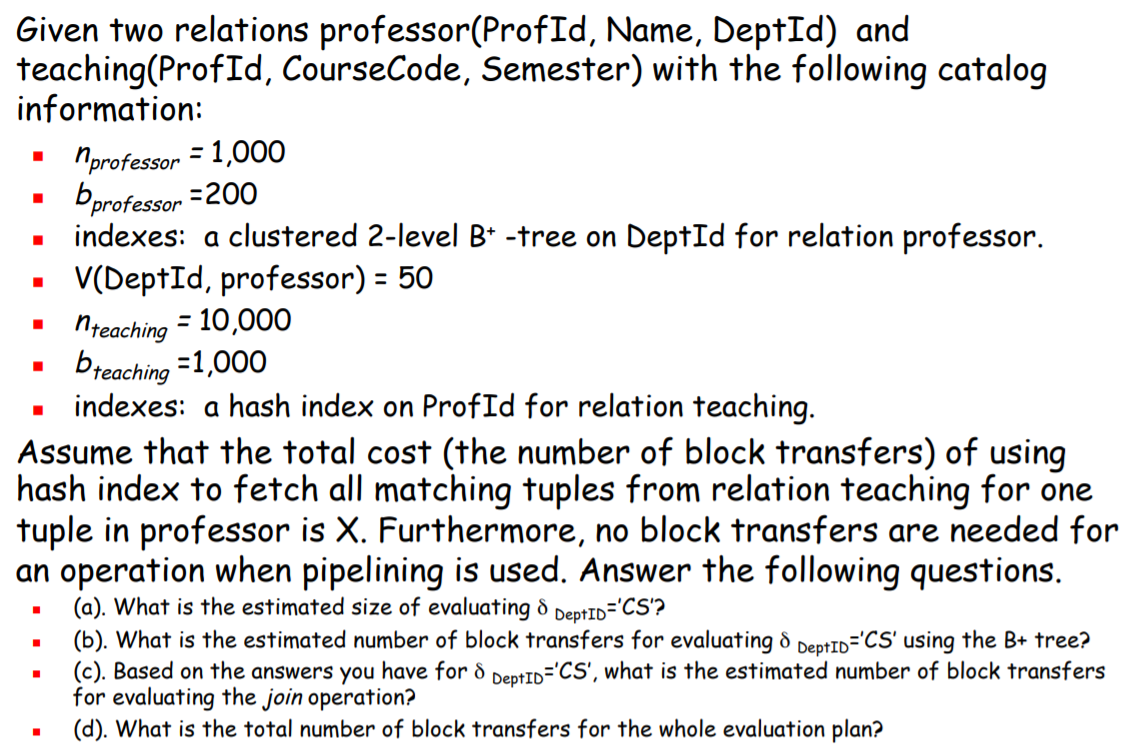
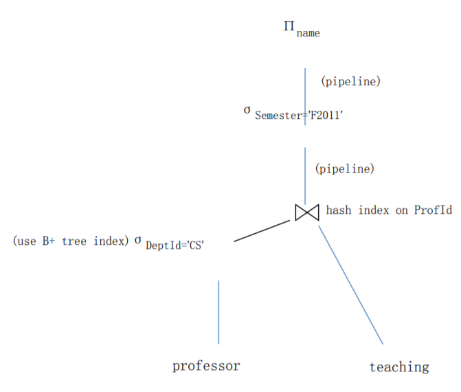
查询优化：

1. 选择运算应尽可能先做
2. 投影和选择应该同时做
3. 将投影运算与其前后的双目运算结合起来
4. 某些关系选择运算+其前面的笛卡尔积🡺连接运算
5. 提取公共因子表达式（公因子）

表达式结果集统计大小的估计：

V（A,r）：关系r中属性A中出现非重复值的个数，如果A是关系r上的主码则V(A,r)等于nr

Ttl5 Q1



a: 1000/50=20 tuples (or 200/50=4 blocks)

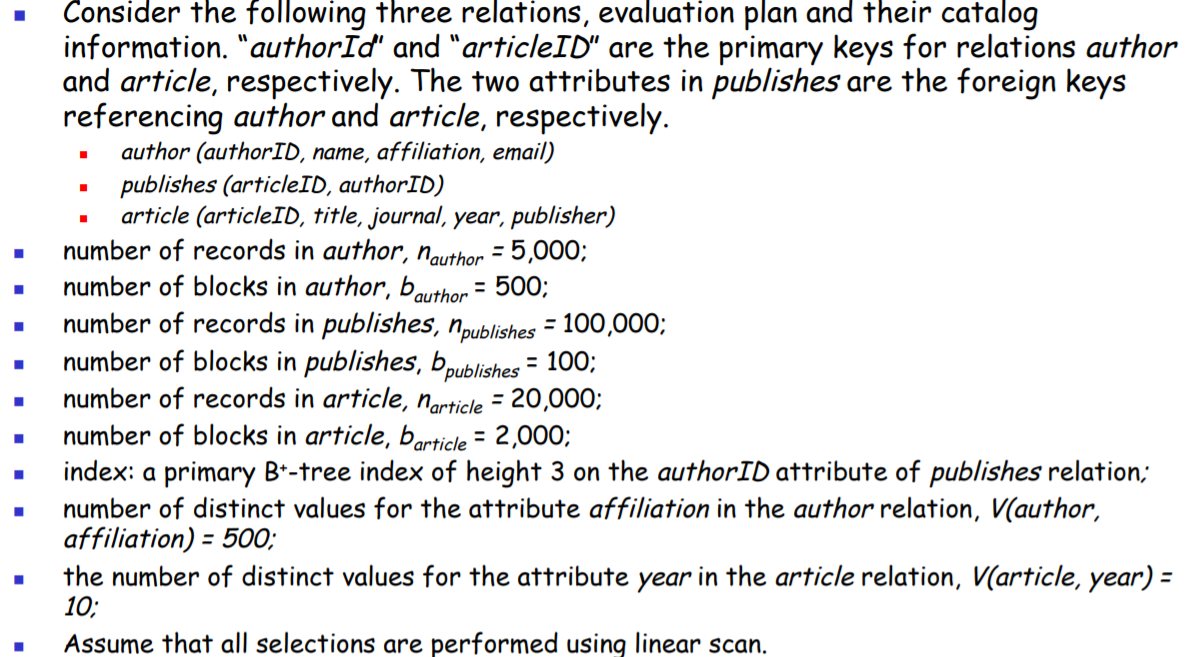
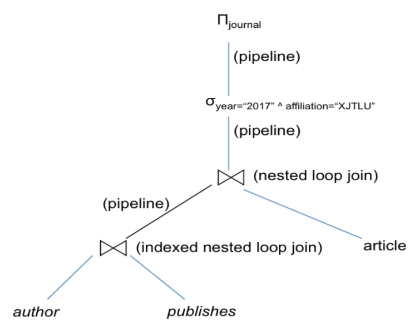
b: 20 records fit into 4 blocks, so the number of block transfer is 2+4 =6

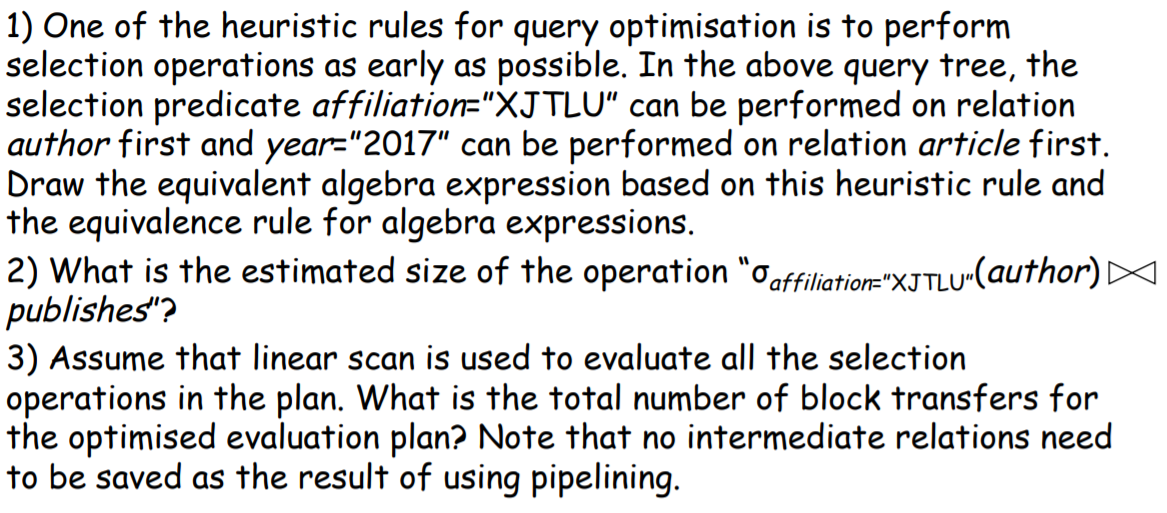
c:因为题目中有给出索引，所以使用循环嵌套连接套用第一部分的公式

br +n\*c = 4+20x

d:if pipelining is used then we do not need any block transfer, but where piplining is not used we have add the cost of storing the intermediate blocks. So the total number of block transfer is 6+4+4+20X

Ttl5 Q2:





2: the above operation can be written as σaffiliation=’XJTLU’(author连接publisher), since autor ID is publisher is the foreign key referenceing author, the estimated size is the number of records in relation publishes, which is 100000. Then we perform th eslection V(author,affiliation) = 500, the size is 10000.500 = 200.

事务：

事务隔离性存在隔离级别，理论上隔离级别包括4个：

第一级别：读未提交(read uncommited)

对方事务还没有提交，我们当前事务可以读到对方未提交的数据。

读未提交脏数据(Dirty Read)现象：表示读到了脏得数据。

第二级别：读已提交(read commited)

对方事务提交之后得数据我方可以读取到。

读已提交存在的问题是：不可重复读

第三级别：可重复读(repeatable read)

这种隔离级别解决了：不可重复读问题

第四级别：序列化读/串行话化读

解决了所有问题。

效率低，需要事务排队。

可串行化：优先图

如果满足以下三点条件的任意之一则Ti🡪Tj

在Tj执行read(Q)之前，Ti执行write(Q)

在Tj执行write(Q)之前，Ti执行read(Q)

在Tj执行write(Q)之前，Ti执行write(Q)

日志文件的内容：

[start,T]: 事务的开始

[write, T, X, old\_value,new\_value]: 事务T更新X

[commit, T]: 事务提交

[abort,T]: 事务撤销

事务故障恢复：事务在运行至正常终点之前被终止

恢复方法：又恢复子系统应利用日志文件撤销(UNDO)此事务以对数据库进行修改。

事务故障的恢复由系统自动完成。不需要用户干预。

事务故障的恢复步骤

1：反向扫描日志文件(即从最后向前扫描日志文件)，查找该事务的更新操作。

2：对该事务的更新操作执行逆操作。将日志文件记录中’’更新前的值‘’(Before Image BI)写入数据库。若是修改操作则用BI代替AI(After Image)

并发控制：

排他锁(X锁)

又称写锁，若事务T对数据对线A加上X锁，则只允许T读取和修改A，其他任何事务都不能再对A加任何类型的锁，直到T释放A上的锁

共享锁(S锁)

又称为读锁，若事务T对数据对象A加上锁S，则其他事务只能再对A加S锁，而不能加X锁，直到T释放A上的S锁

一级封锁协议：在写之前加上X锁，写之后释放X锁

二级封锁协议：在读之前加上X锁，读之后释放X锁，此时的问题是两个读之间有空窗期

三级封锁协议：在读之前加上S锁，读之后释放S锁，在写之前加上X锁，写之后释放X锁