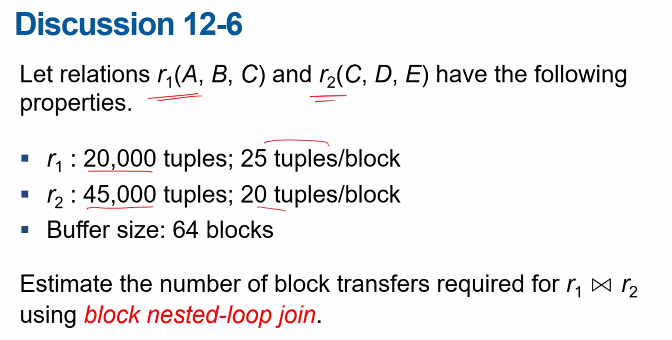
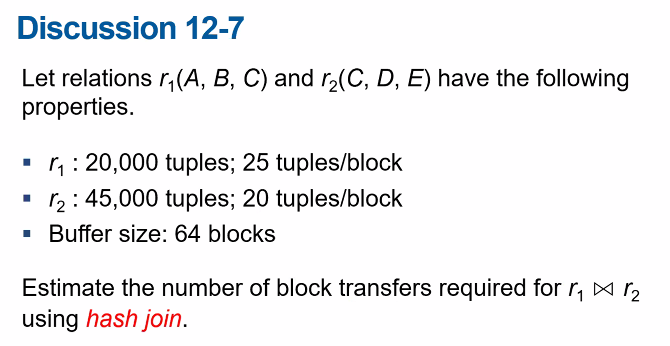
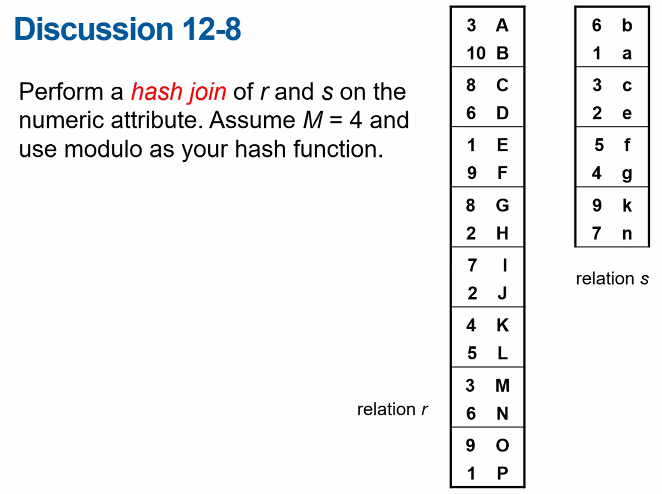
**Discussion 06/01**



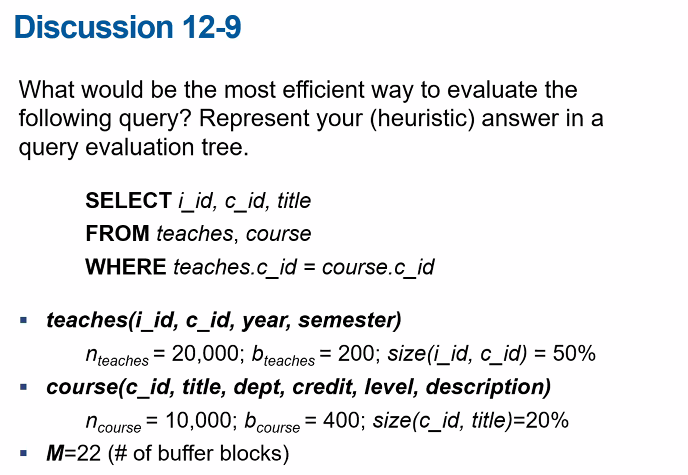
r1의 block 수는 800, r2의 block 수는 2250. 버퍼 수가 64개이므로 Cost를 계산하면 [800 / 62] \* 2250 + 800 혹은 [2250 / 62] \* 800 + 2250. 두 가지 경우 중에 더 작은 걸 택하면 될 듯.



3 \* [800 + 2250] + 4n\_h인데 n\_h는 파티션을 몇 개로 하느냐에 따라 다름.

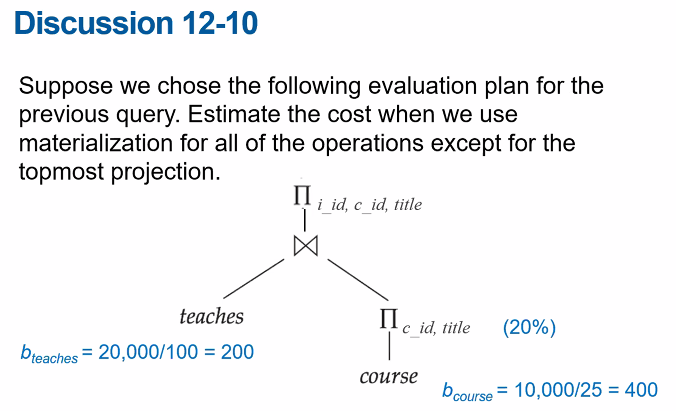


=> M은 보통 꽤 큼. Partition 개수보다 일반적으로 memory buffer가 더 큼. 적절한 partition 수는 어떻게 정하는 것이 좋을까? 작은 table의 partition된 각 block들이 메인 메모리 buffer에 완전히 들어올 수 있게 만들어 주는 것이 좋음. 따라서 각 파티션이 2개였으면 좋음.



제일 먼저 두 개의 테이블 각각에 대해서 projection operation을 먼저 수행하면, size가 teaches는 50%, course는 20%로 줄어든 상태에서 join을 하기 때문에 효율적일 것 같고, 그 join 연산의 결과는 pipelining을 해서 그대로 사용해서 위의 select operation을 수행하면 좋을 것 같다. 이 때 c\_id를 알아야 하기 때문에 두 relation 모두에서 c\_id를 뽑았음.

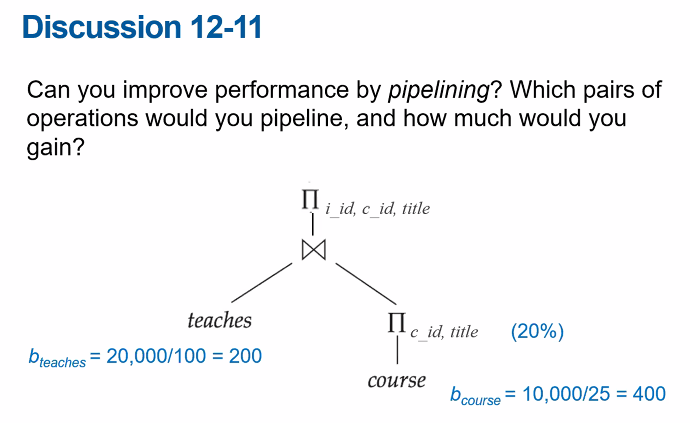
즉, 최하위에 project\_(i\_id, c\_id)\_(teaches) 와 project\_(c\_id, title)\_(course)가 있고, 걔네 두개가 join으로 합쳐지고, 그 위에 select\_(teaches.c\_id = course.c\_id)\_(join의 결과) 이렇게 나올 것 같음.



course에서 projection 할 때 400블락 읽고 80블락 저장.

그럼 teaches와 join하면, 간단하게 hash를 쓰면 3\*(200+80) = 280\*3 = 840 block transfer.

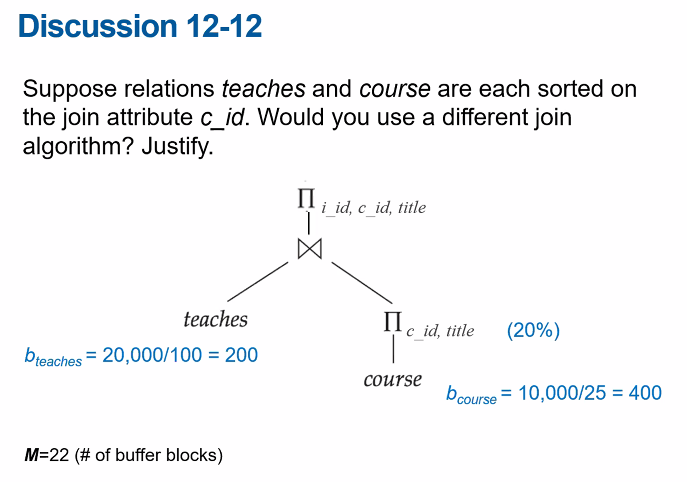
Join 후의 block 개수는 200 + 80\*2 = 360개.



Join 후 projection할 때 pipelining이 가능. 또 course에서 selection을 한 결과를 pipelining하면 80개의 block을 쓰지 않아도 되지 않을까?

=> 근데 M이 22이므로  이런 식으로 20개씩 나눠서 읽어들여서 그 때마다 teaches를 읽는 식으로 해야 함.

그럼 join cost가 [80/20] \* 400 해서 800임. 따라서 전체 cost가 1200.



=> 두 개의 relation에 cursor를 하나씩 두고 two pointer algorithm을 사용해서 **MERGE JOIN**을 함. Potentially 200~400 개의 block transfer. Merge join은 join attribute에 대해 sort가 되어 있어야 함. Duplicate value에 대해 처리를 해줘야 해서 Buffer 크기를 크게 잡음. 그래서 교수님이 반반으로 계산.

