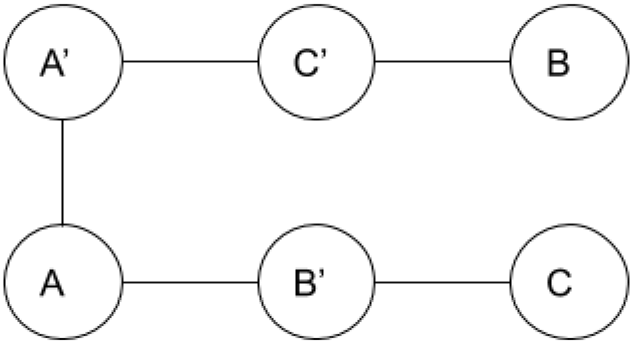
# CS386D Database Systems
# HW9a & b Solutions

## 9a. Part 2

**Note:** These answers are for using Postgres 11.5 on AWS RDS. Answers may vary slightly depending on the DBMS configurations set.

A) Merge-join chosen for SELECT * but not for SELECT A.pk, B.pk.
B) Hash join (after sequential scan) was chosen for both queries.
C) No impact even for SELECT A'.pk, B'.pk
D) Merge join was chosen on rows sorted by join attributes A.ten, B.ten. Note: Even though this is similar to B, a hash join here would result in large partitions given V(R, ten).
E) Merge join was chosen without the need for sorting.
F) The optimizer chose its own order no matter the order in the query. This may follow from the optimization of the number of buckets created during hash joins.
G) Same as F.
H)



I) The estimated result should have ~3*$10^{14}$ rows (may vary of course).

**9b.1)** Option B.

**9b.2)** Apart from just the ranking, it is important to notice the trend of use. It has a more or less consistent graph. We can speculate the reason as being legacy systems still dependent on the system that cannot be revamped easily. I wouldn't call it an important technology but it may still last a while (this is a matter of opinion but should be supported with reasons and statistics).

### A note from Professor Miranker:

*Dear Class,*

*While I still leave the grading to the TA, for this particular homework I like to review a number of papers, and the physical query plans created by the optimizer. I use that to make changes and improve the assignment as time goes on.*

*This semester, there is a difference in plans that surprises me. That each new version of PostgreSQL emits different plans than the previous version is not a surprise. The surprise with the current version is in Query 1,*
*Recall this is the query I refer to as "the golden case" for merge-join, two tables joined on their primary keys. The rows come off the disk presorted on the join argument making enabling the merge in a single sequential I/O access.*

*In previous versions of PostgreSQL the optimizer did choose a merge-join.*

*Not an actual assignment - what do you suppose might account for this? If you looked at the two versions in detail, what would you look at to validate or invalidate your hypothesis?*

*DM*