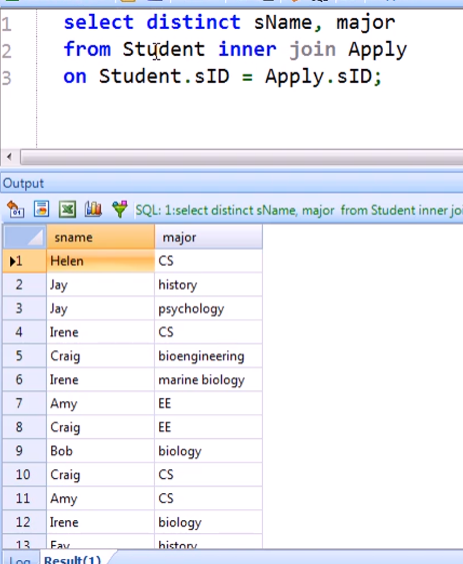
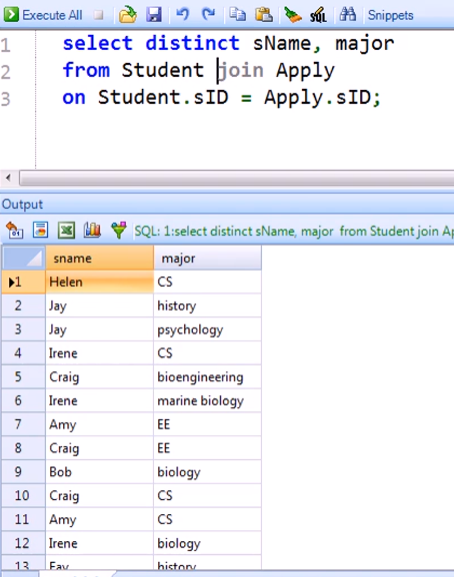
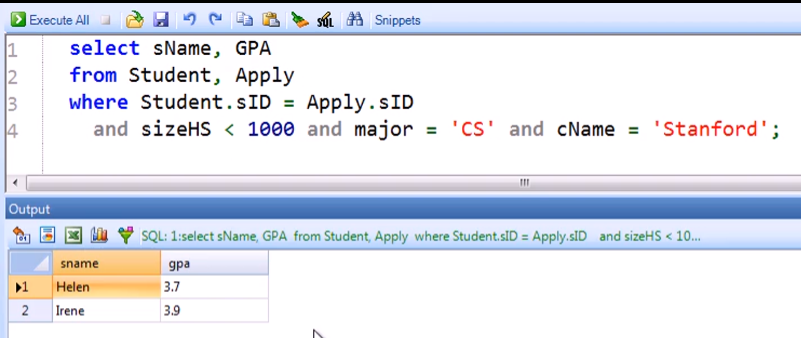


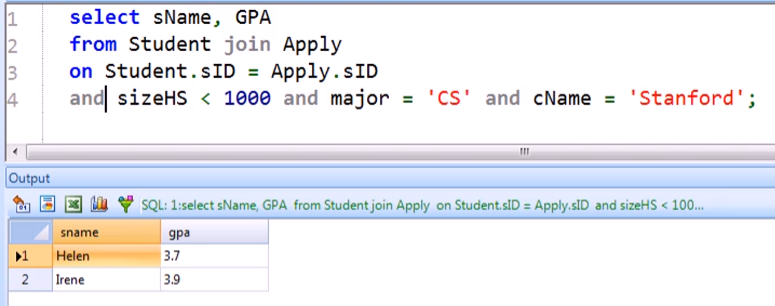
INNER JOIN

* This is like the theta join
* Default as “join”

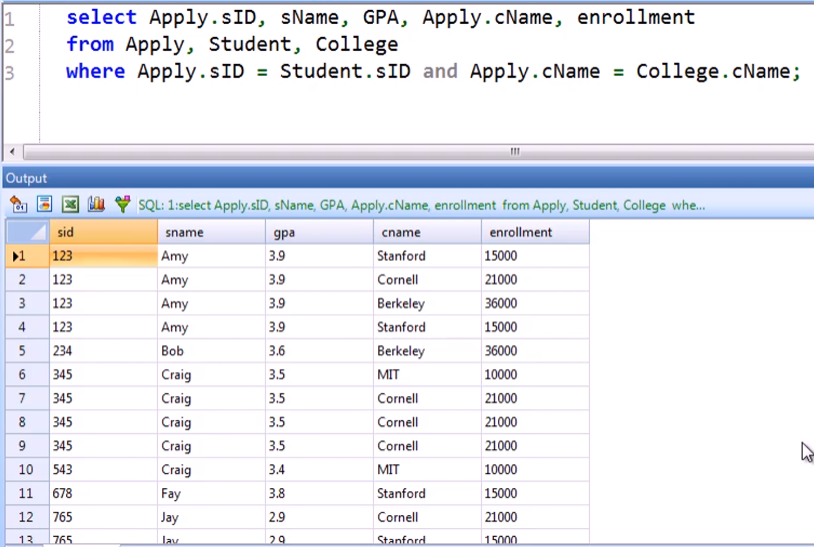




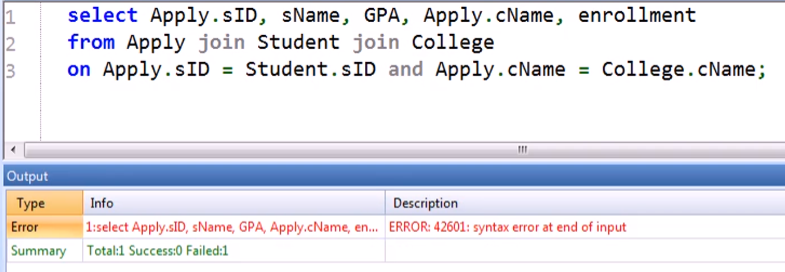


JOINING THREE RELATIONS

Comma form

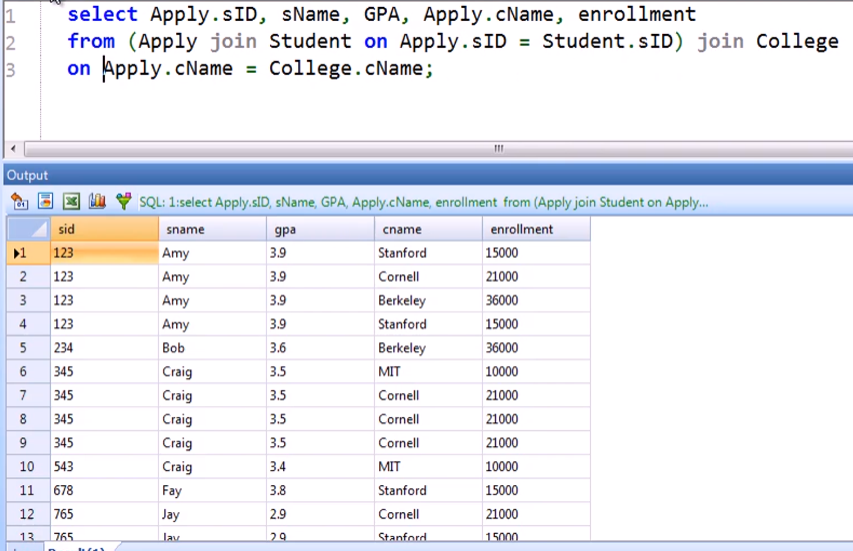


Using JOIN



* Three-way join is not supported by PostgreSQL

Binary joins only for PostgreSQL

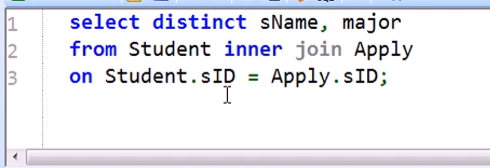


* Join first the 2 tables including its condition

NATURAL JOIN

* Automatically joins on the same values on common attribute names without having to explicitly indicate on where the columns would match

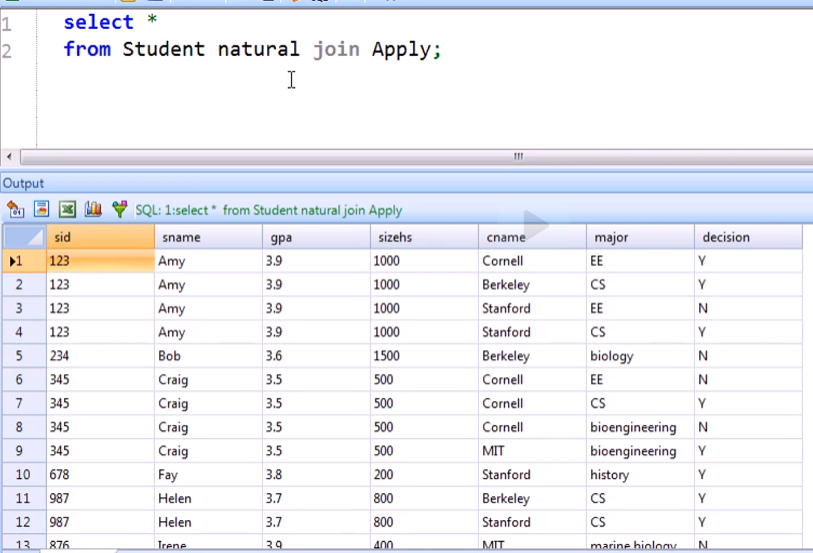
From here:

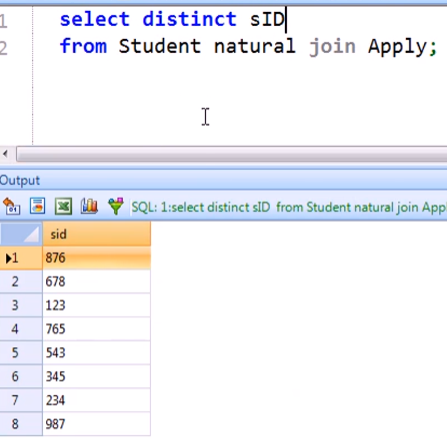


To here:



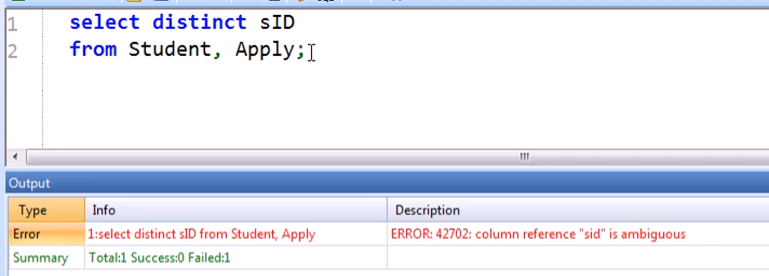
Also retains only 1 column of the same relation:





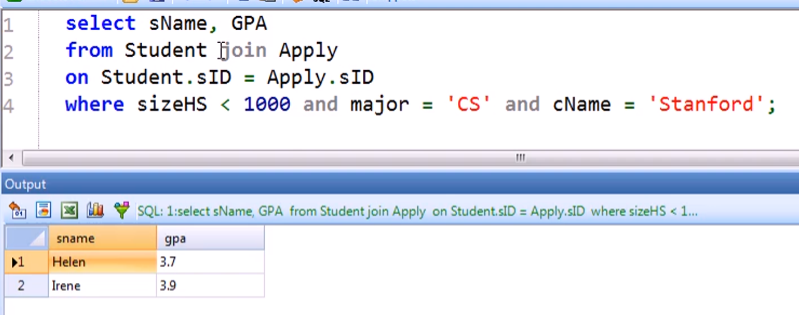
* 1 column of SID only

Without NATURAL JOIN:

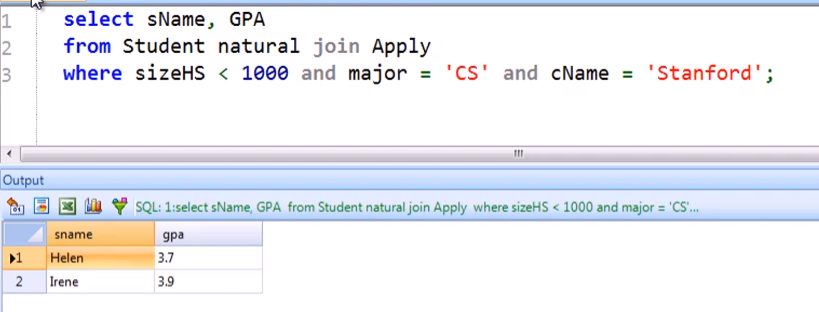


* Have ambiguity
* Need to explicitly indicate

Another example:

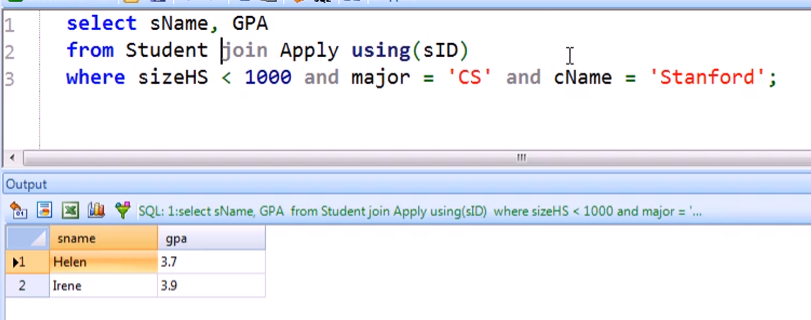


Using NATURAL JOIN



USING clause

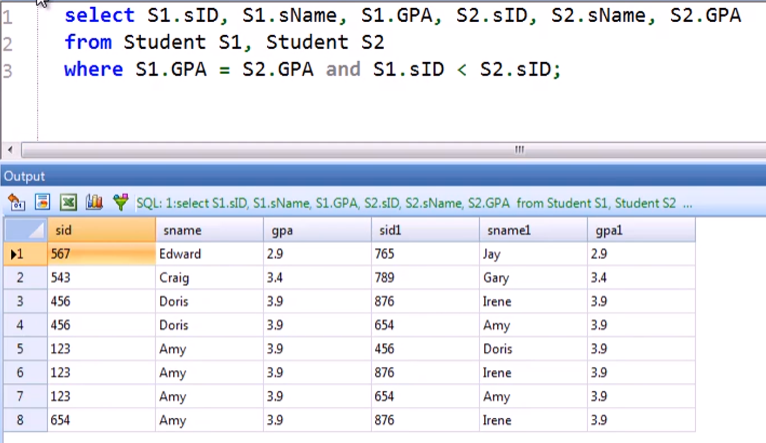
* Explicitly indicate what column/attribute should be equated by the JOIN



* Better practice than NATURAL JOIN

Another Example with USING

Normal Query

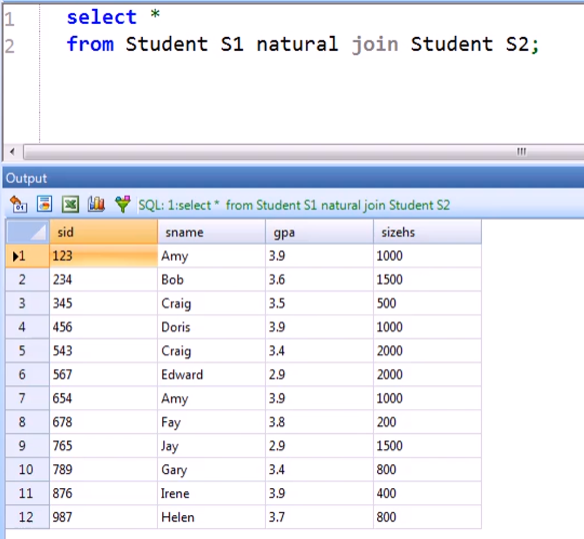
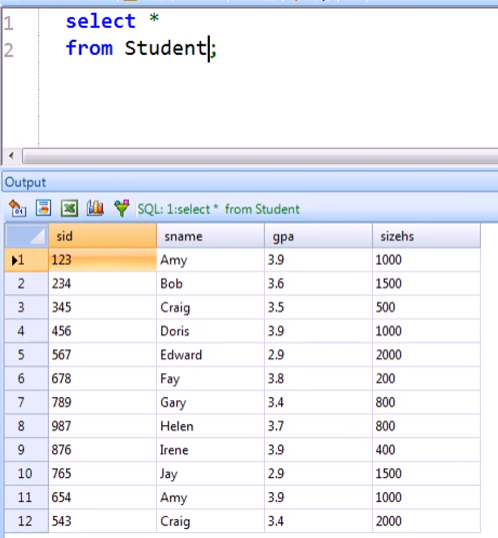


With USING and ON clause



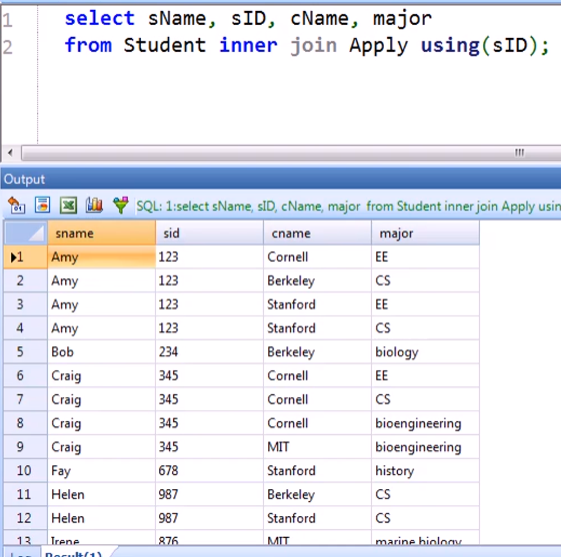
* Most system does not allow both USING and ON at the same time
* You can only use one when using join

Self-joining using NATURAL JOIN

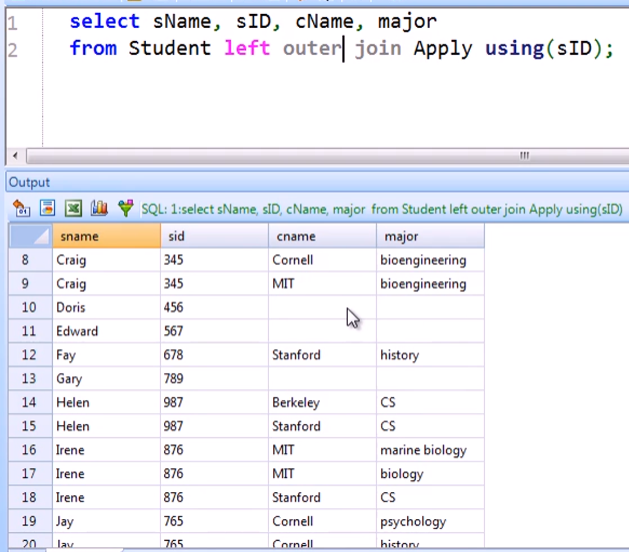
OUTER JOIN FAMILY

Starting Query: Inner Join



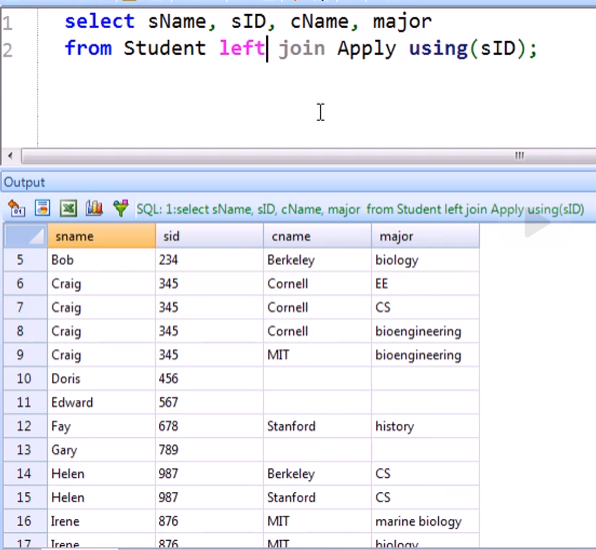
If we want to include students who have not yet applied to any college and major, just replace inner with **left outer join**

* This will include all the students with or without values (null) in the Apply table

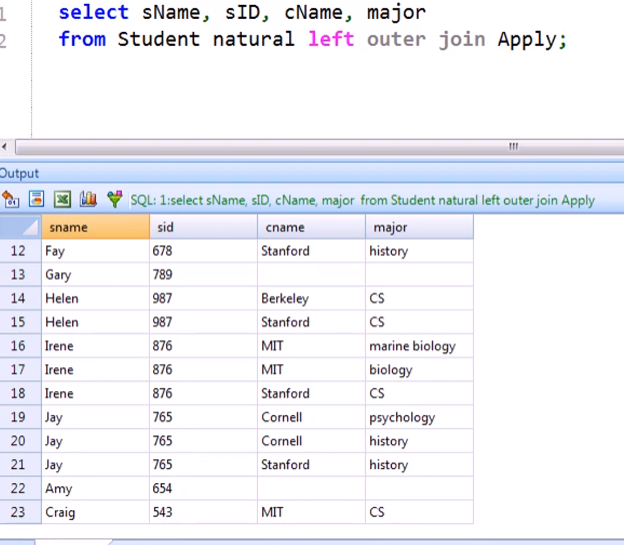


* Tuples with no match on the 2nd table are called “dangling” tuples

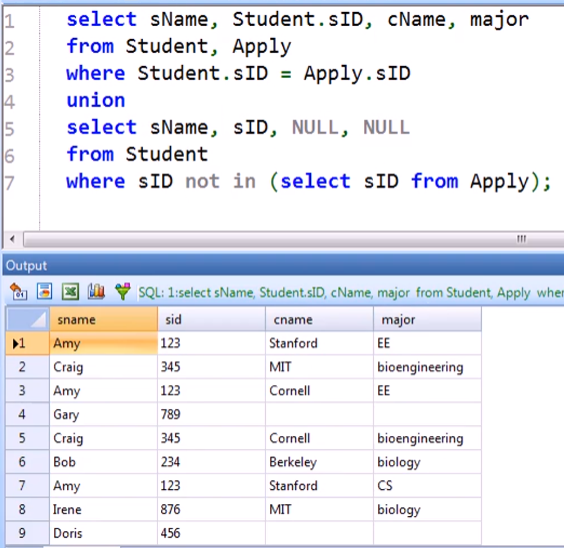
You can also use simply “left join”



We can also use “NATURAL LEFT OUTER JOIN”

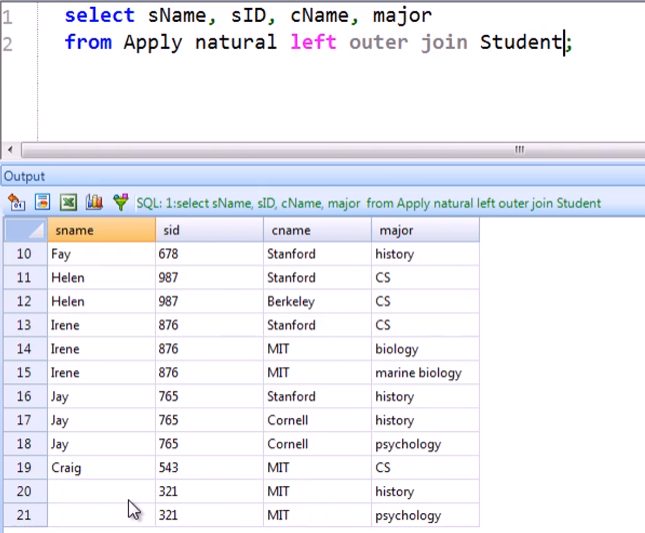


Rewriting the left outer join without using “left outer join” explicitly

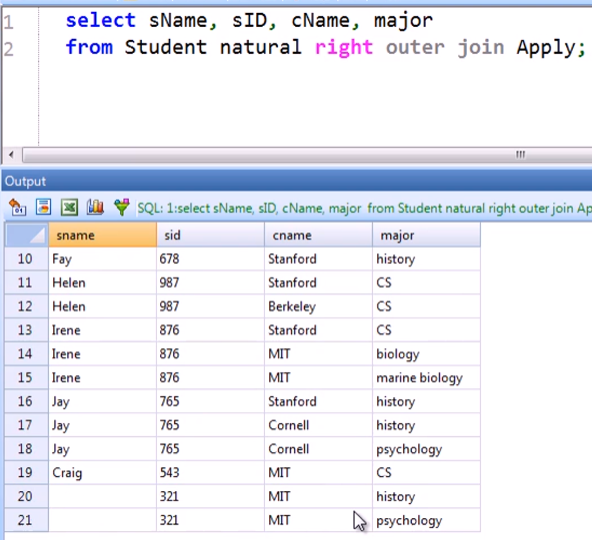


What if we want to retain the Apply tuples (Apply tuples as the dangling tuples) instead of the student?

By Swapping:

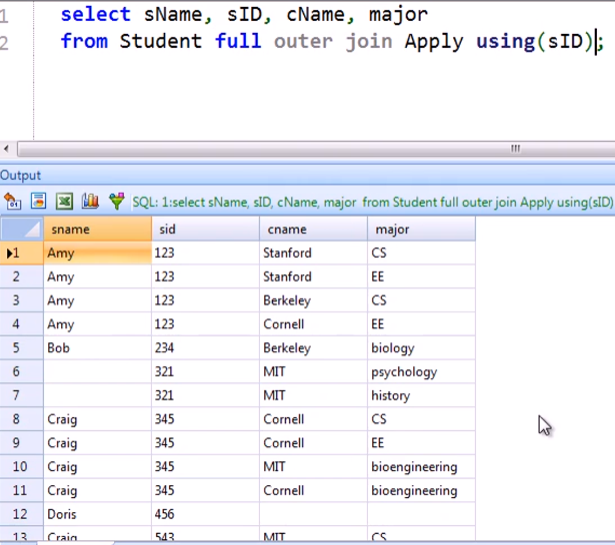


By just using “**right outer joi**n”

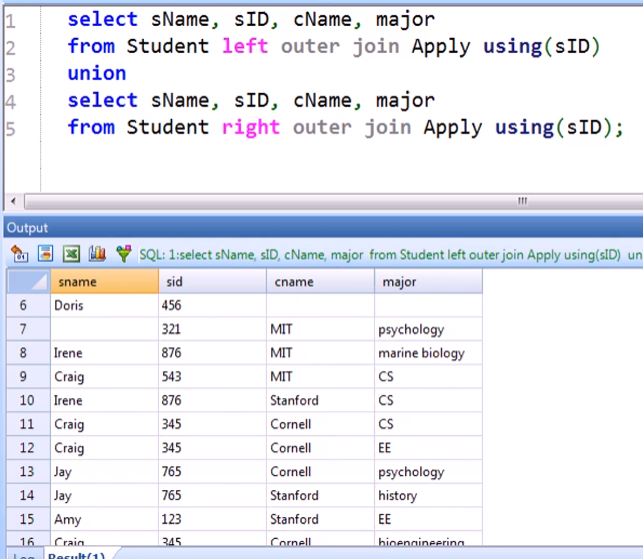


What if we want to retain BOTH SIDES OF UNMATCHED tuples?

Using “full outer join” and USING

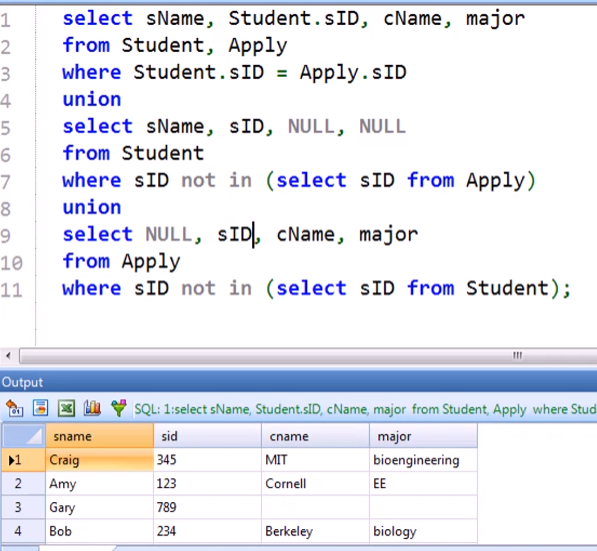


Rewriting FULL OUTER JOIN using LEFT OUTER JOIN and RIGHT OUTER JOIN

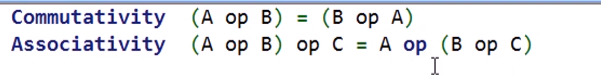


* Remember: UNION automatically remove dupes

Rewriting FULL OUTER JOIN without using any JOIN family

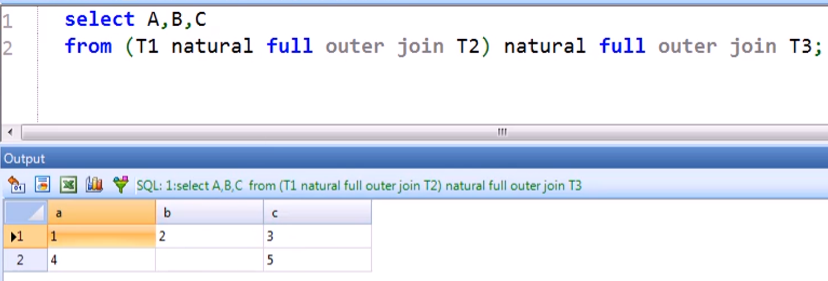


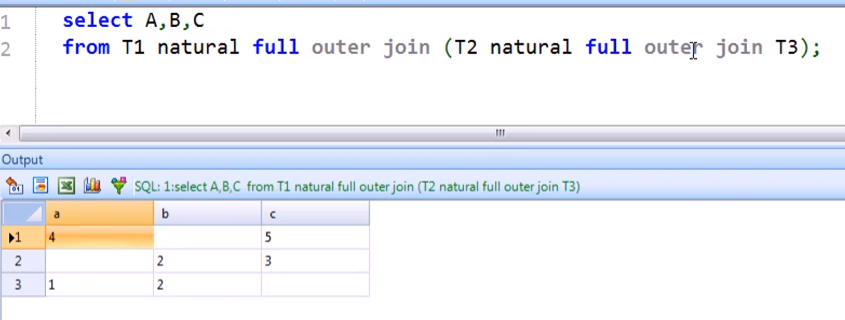
Pitfall of the OUTER JOIN



* This is an exception for OUTER JOIN!

NON-ASSOCIATIVITY OF THE OUTER JOIN:





* FULL, LEFT, AND RIGHT ARE NON-ASSOCIATIVE

NON-COMMUTATIVITY OF OUTER JOIN

* ONLY LEFT AND RIGHT ARE NON-COMMUTATIVE