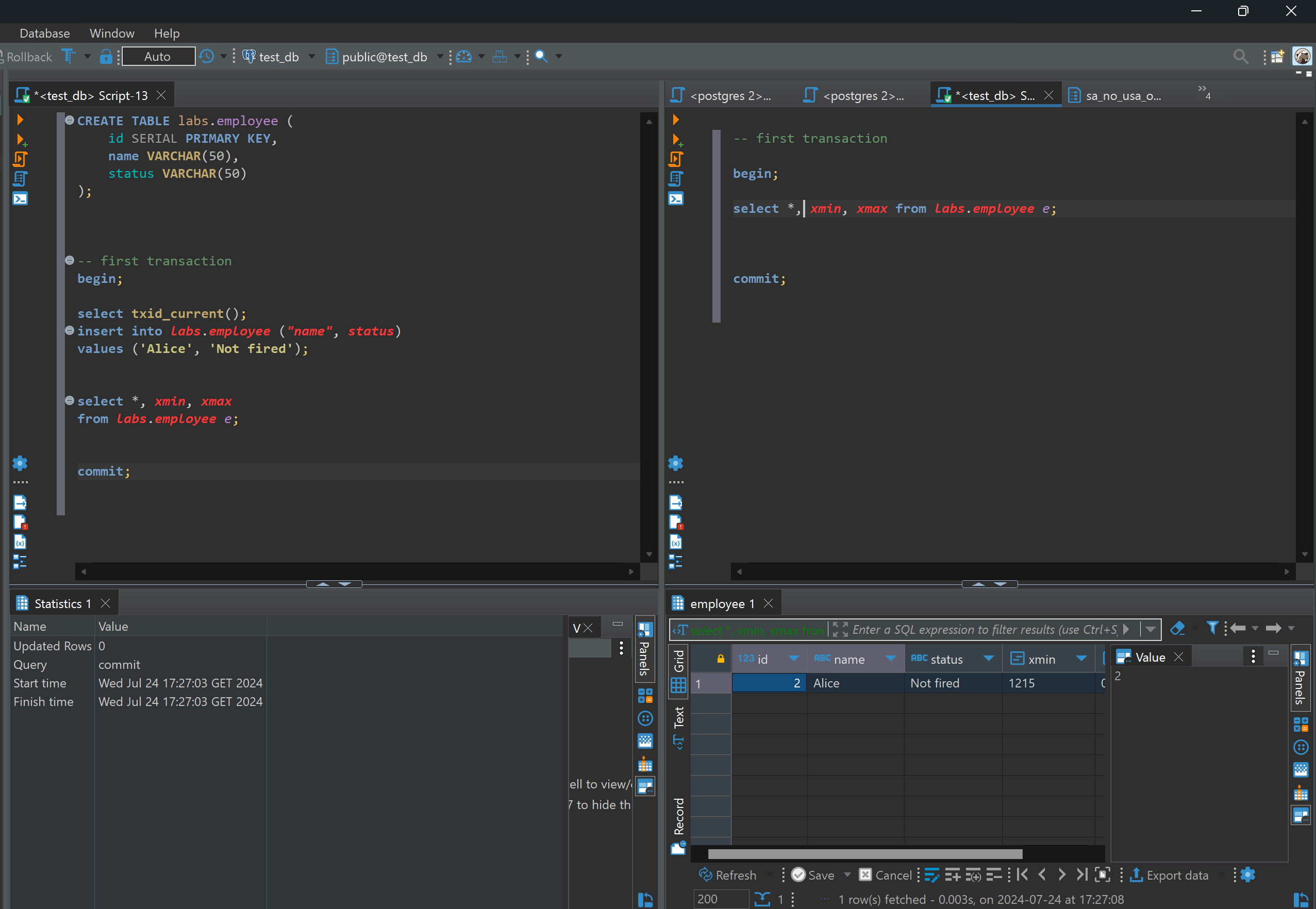


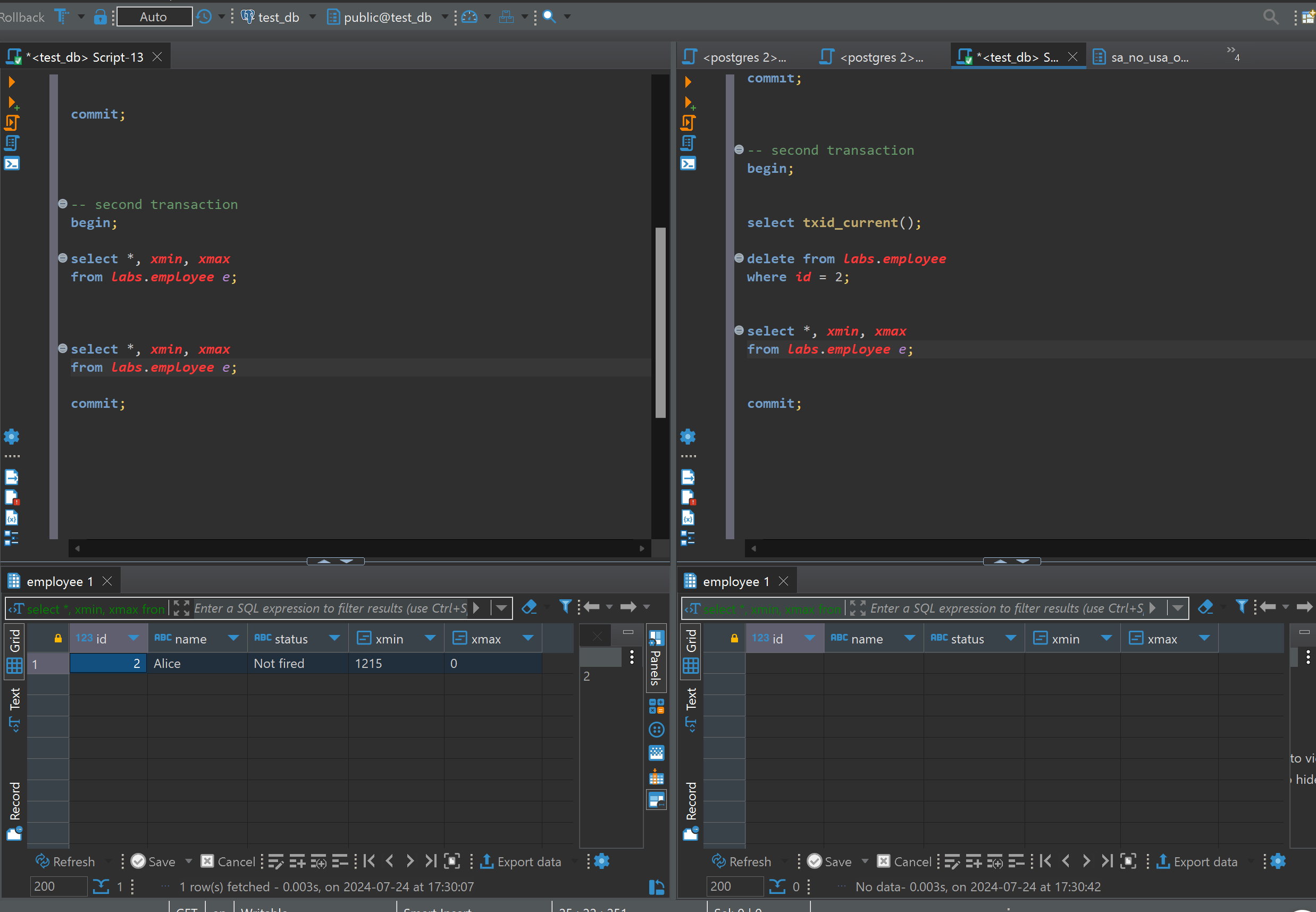
First I ran transaction on the left side, I run it, but It is not committed as I can se xmin, inserting value is 1215.

And while first transaction is not committed, the right transaction select shows nothing.



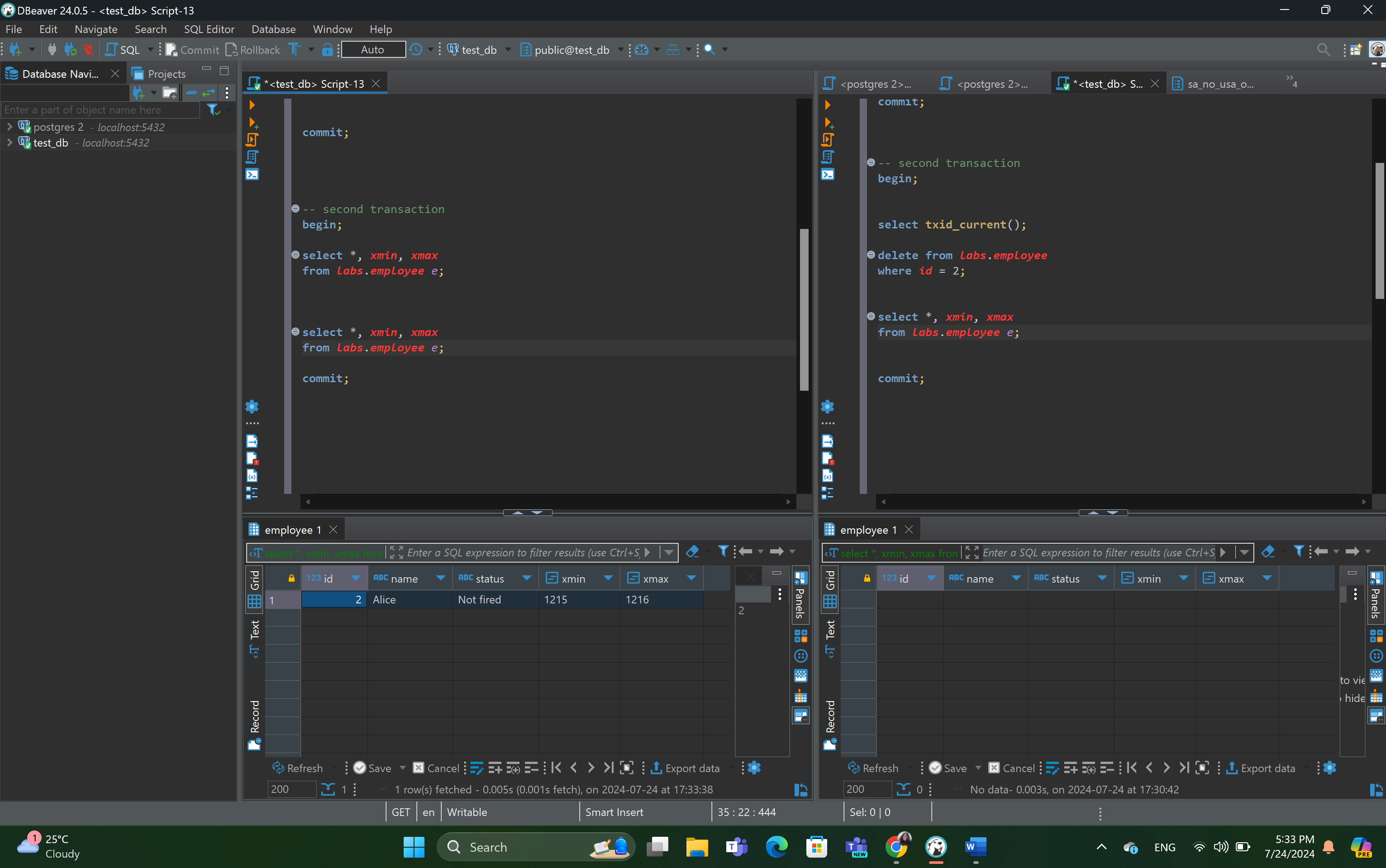
After committing left transaction, right transaction select works and shows Alice.

Second transaction

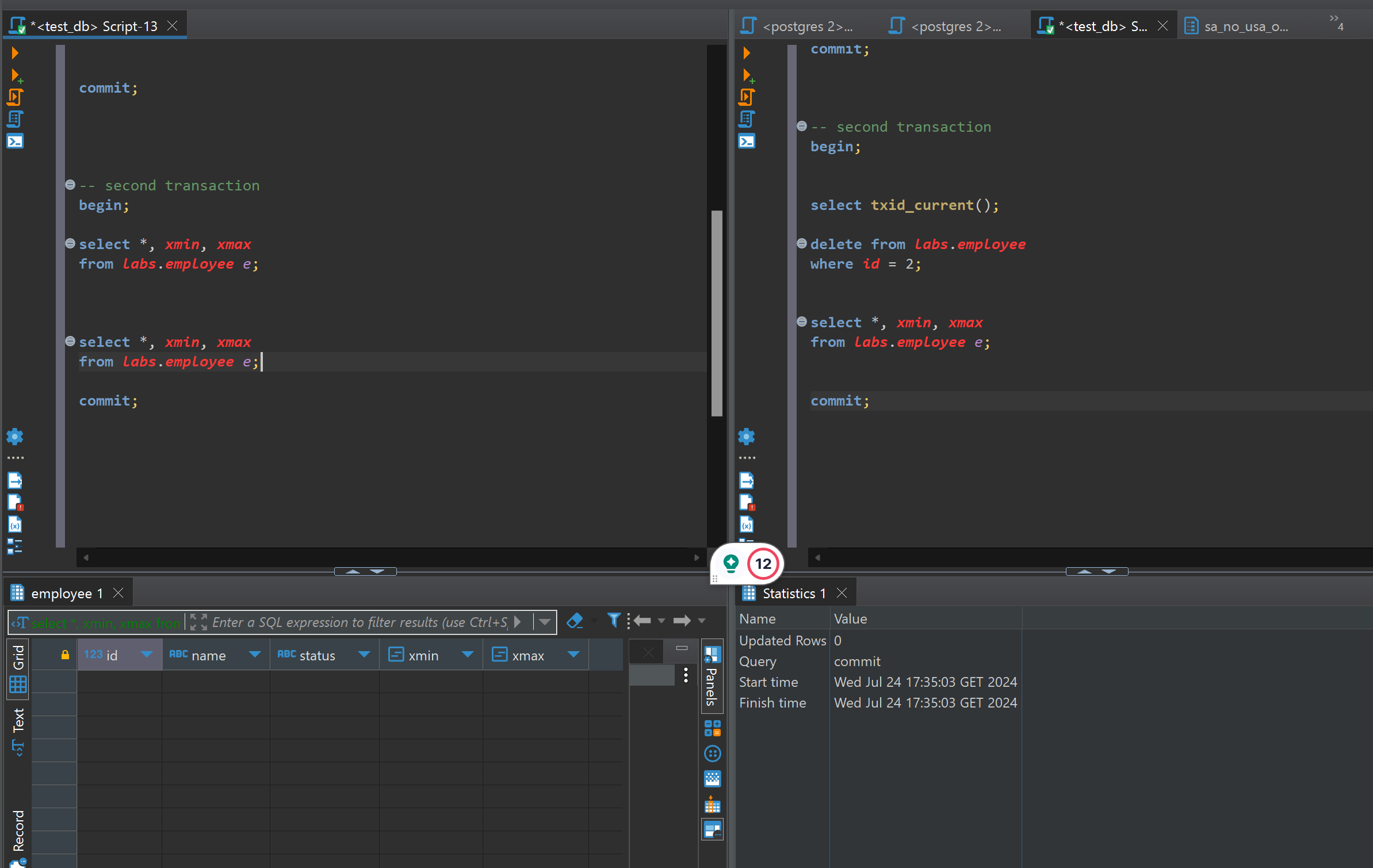


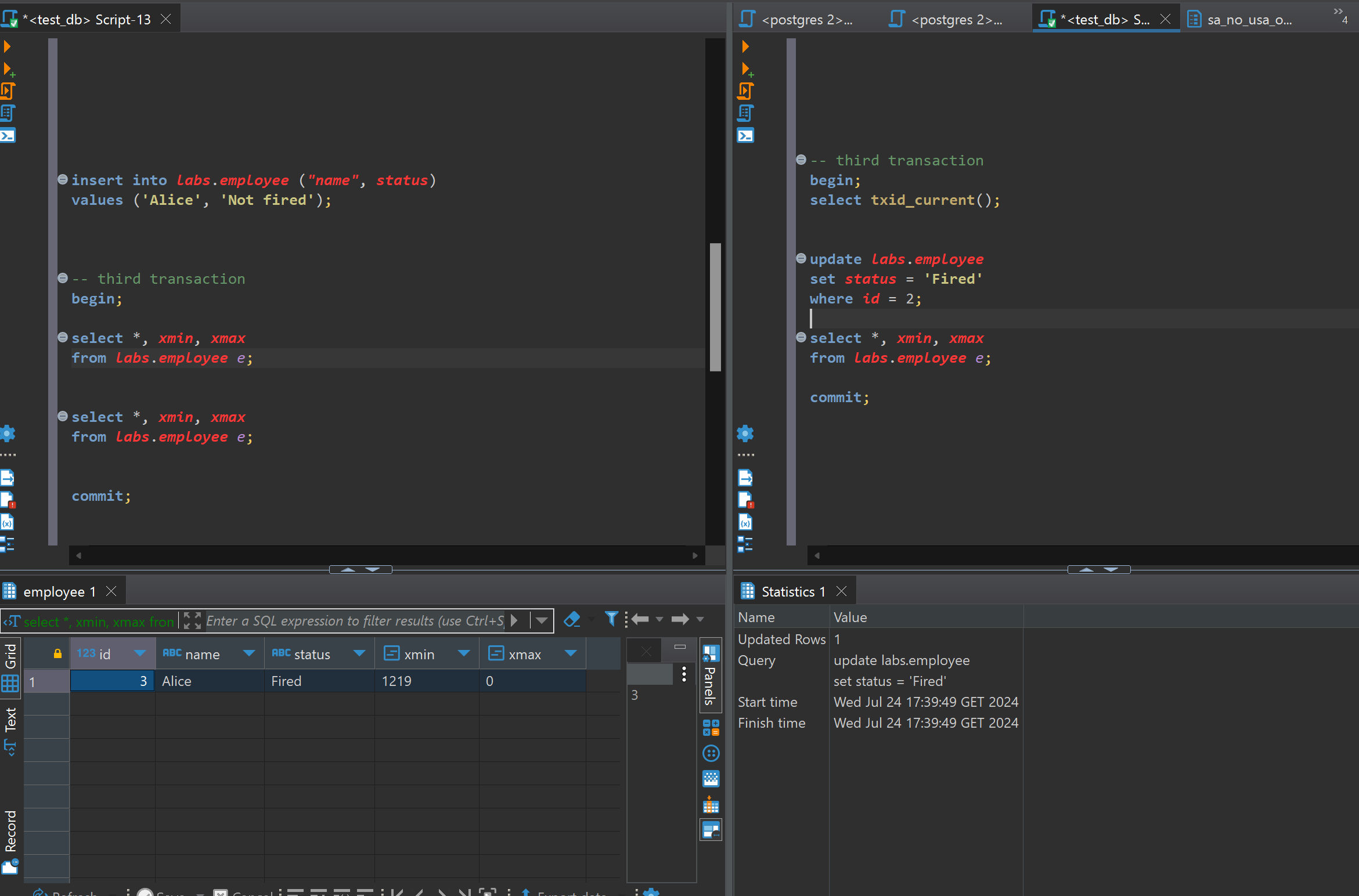
After running left transaction first, it is obvious that nothing is changes, transaction insert id – xmin is same,

But while it shows that right transaction deleted Alice and when selecting its gone, while in the left statement it is still visible.



After deleting and running left transaction select, it is visible that xmax has become 1216 as it is delete id. And it means that in right transaction Alice is deleted.

 after deleting transaction is committed the select statement does not work on the left, because delete is committed. It does not exist anymore.

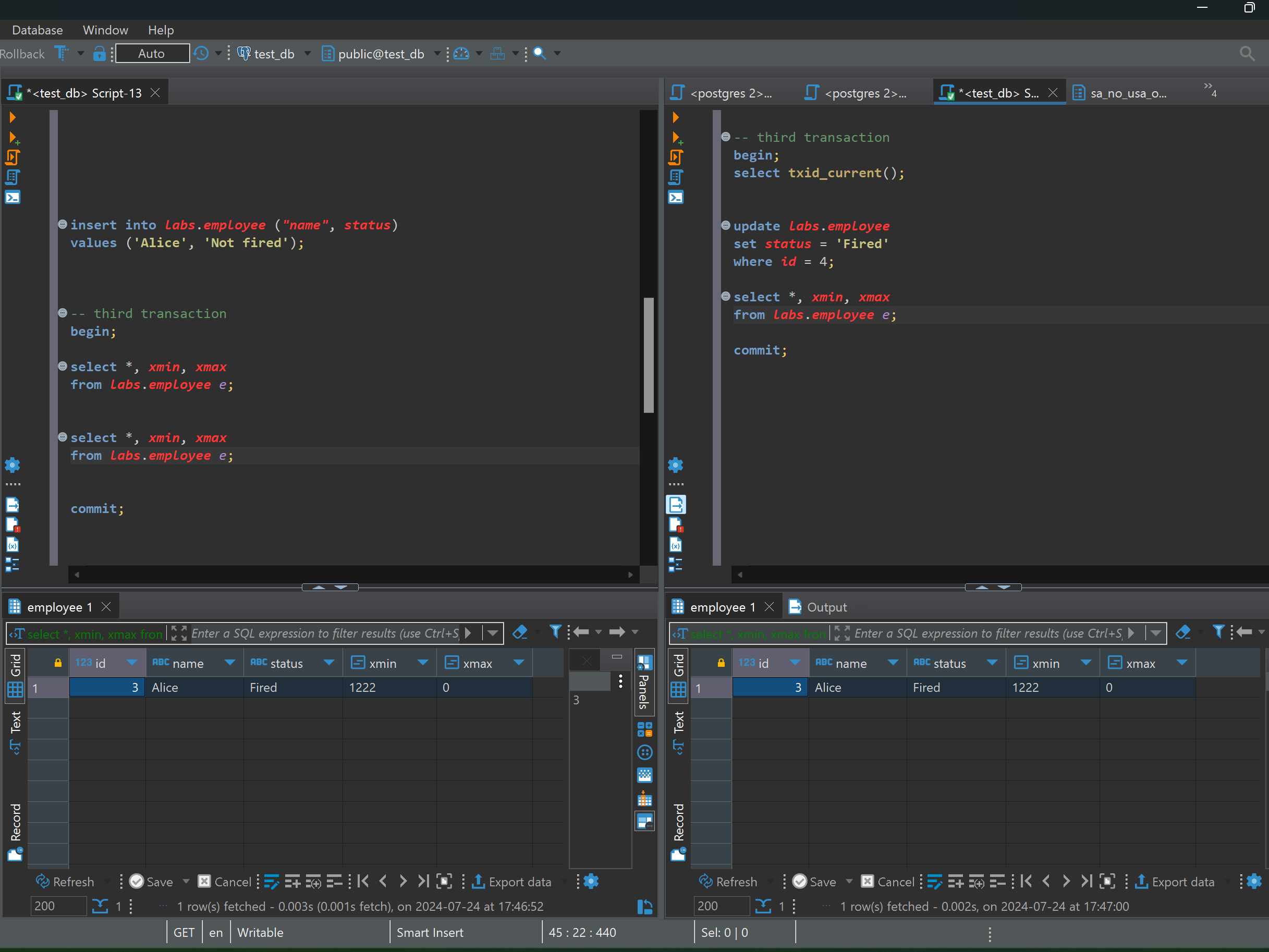


In third transaction. First, I ran left transaction and select and then on the right I deleted the transaction.

Somehow after update and running select on the left side, the ‘not fire’ updated to ‘fire’

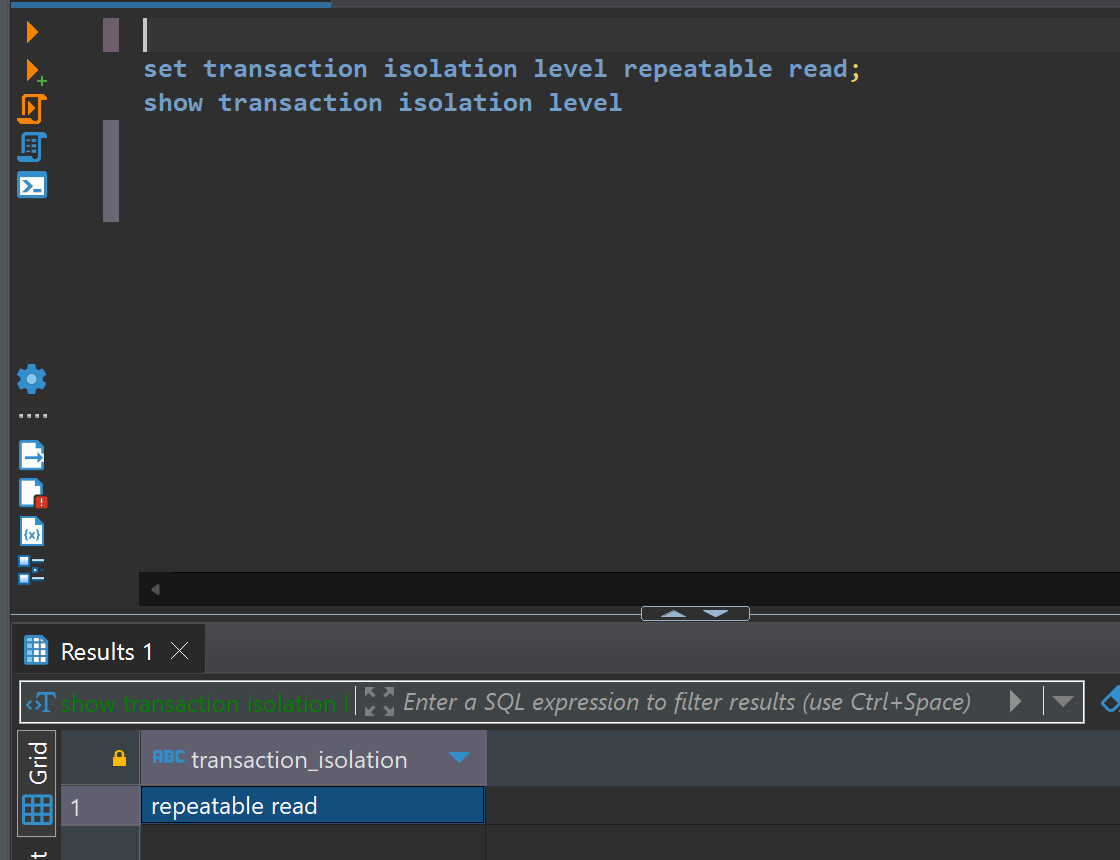
And xmax does not change at all

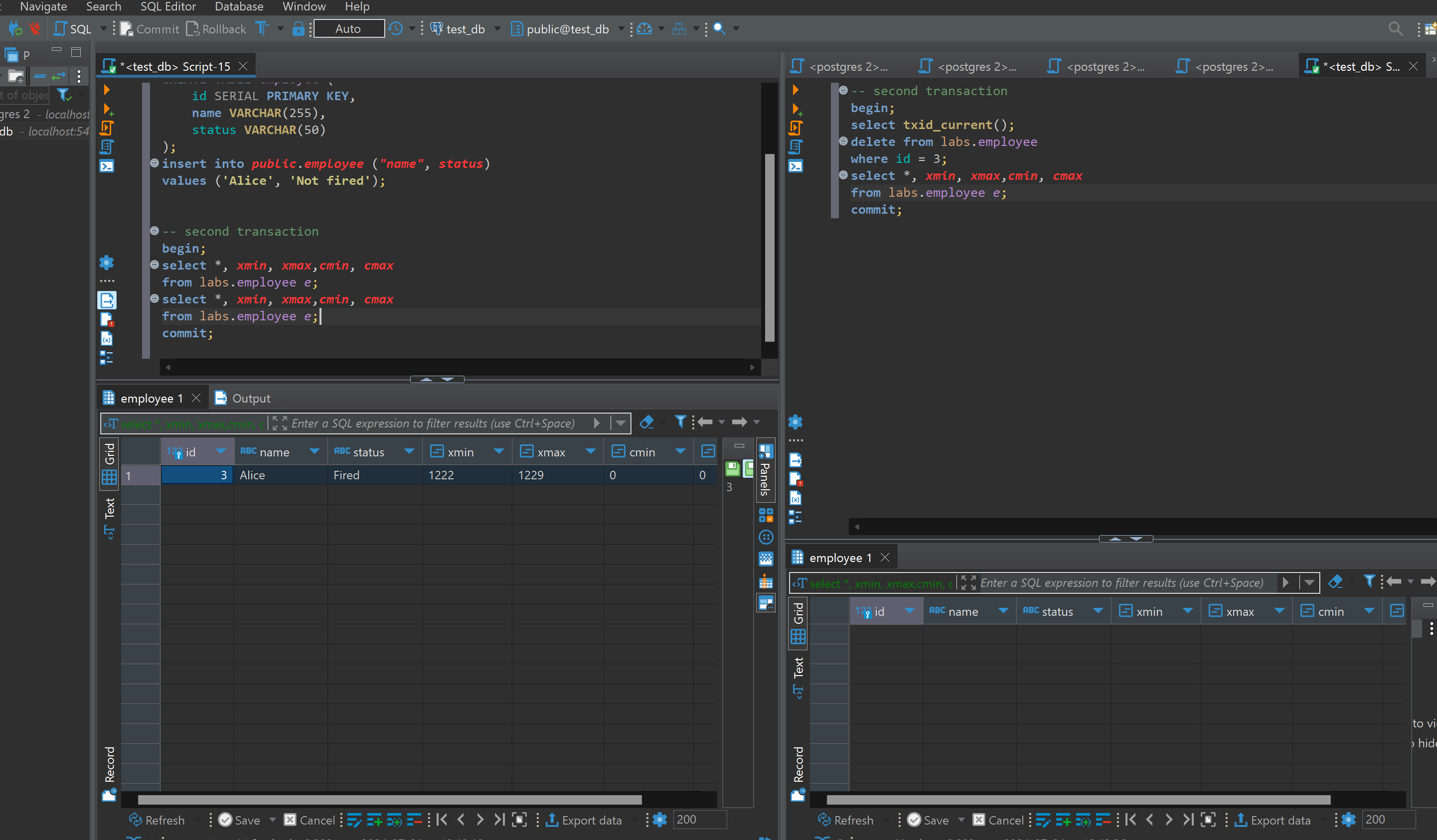
.



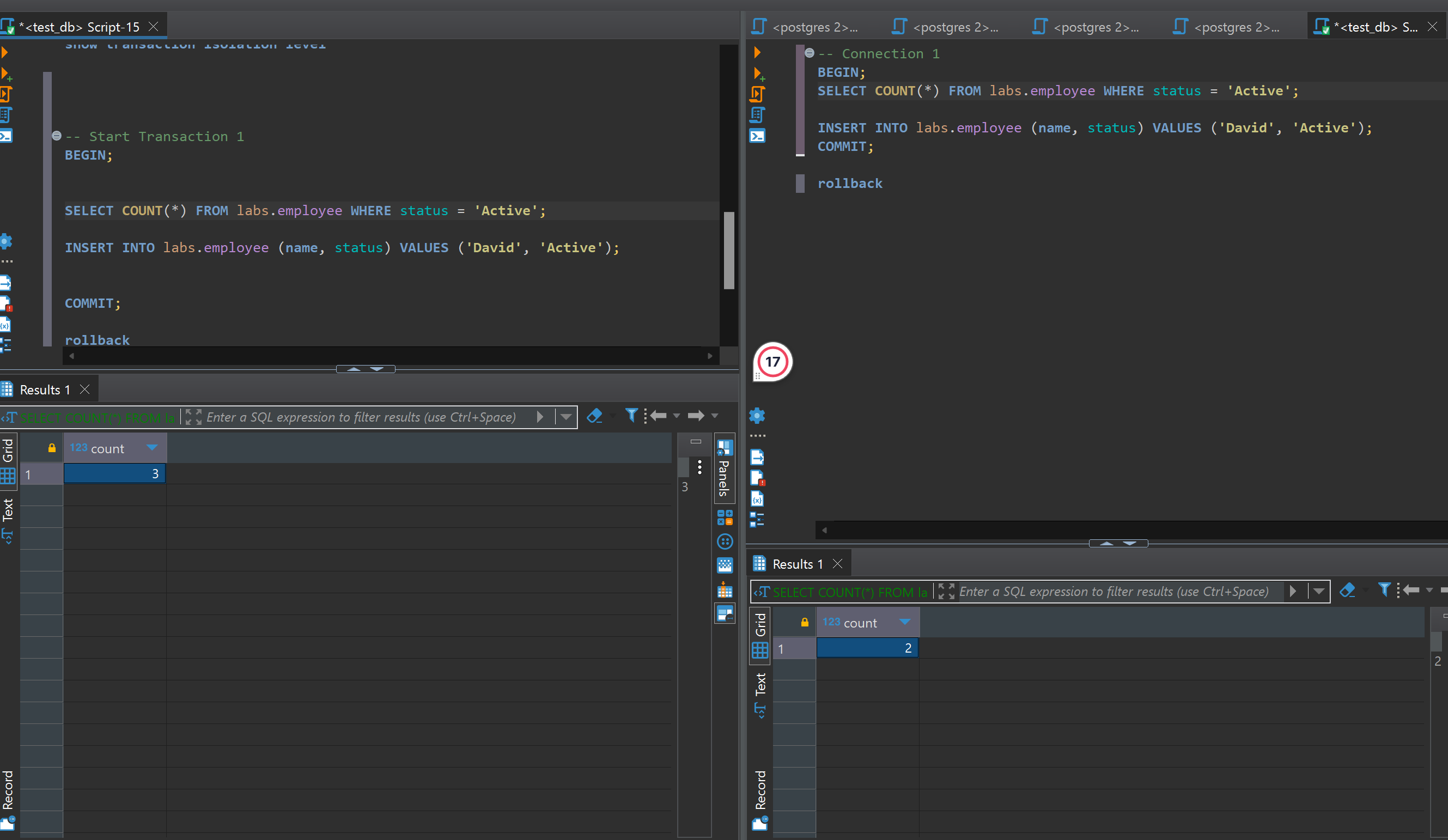
Adding cmin, cmax;

Changing isolation level to repeatable read.

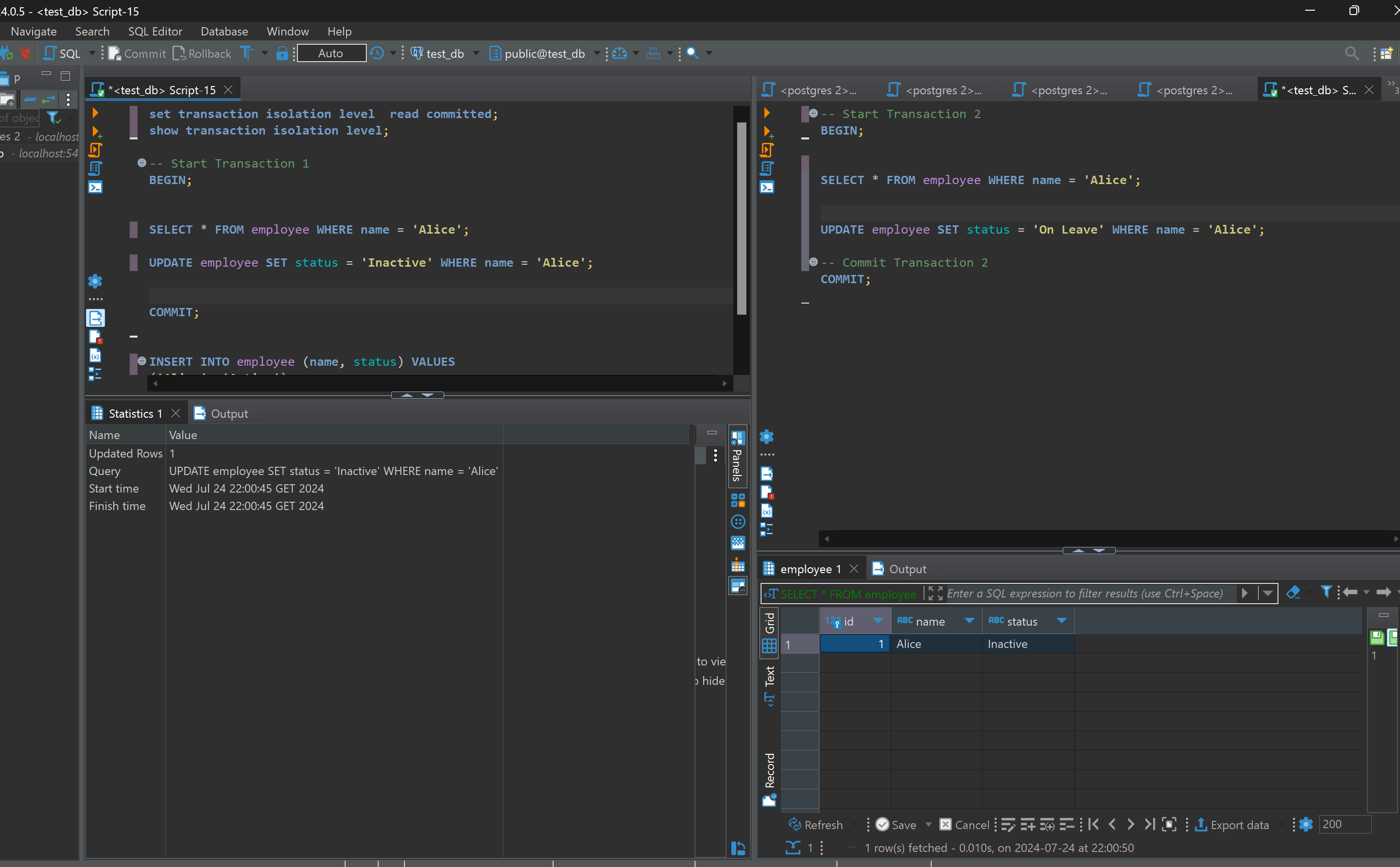




Nothing really changed



I made isolation level serializable and as I can see, in the left transaction at the beginning I had like 2 active and then after inserting one more insert, then i have 3 actives, and in the right side, even though I updated table I still have 2 actives.and after David will be added twice.



Even its uncommitted change, it has shown on the right side that alice become inactive,

It read uncommitted changes. While it is not commited, other transaction can see it and it Is not good, because it violates consistency. Expected outcome will be that one of the updates will be lost because both transactions read the same initial state and then write their changes without knowledge of the other transaction's update.

Here update on leave stays but other updates lost.

