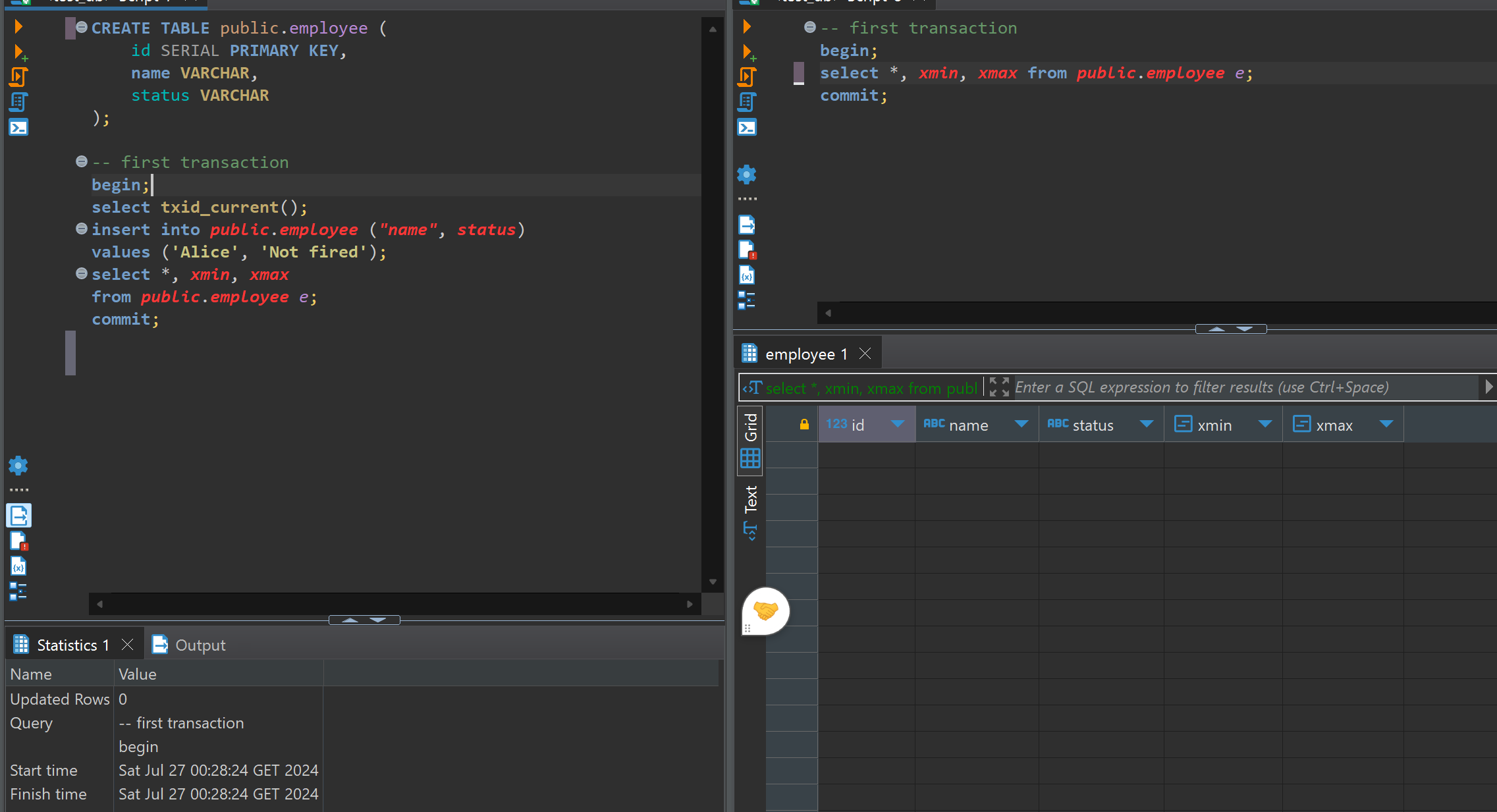
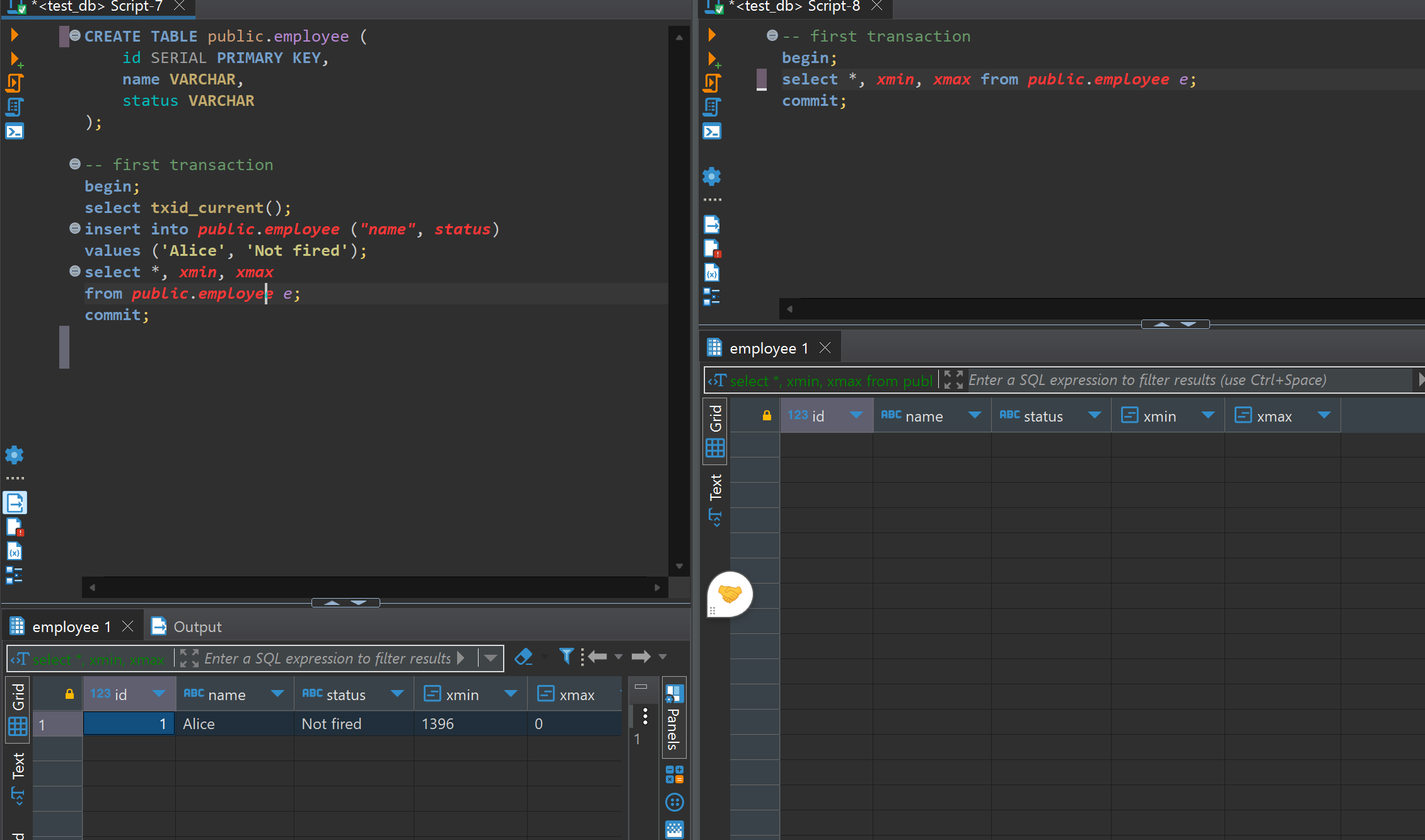
1. Created table
2. Executing transaction 1:

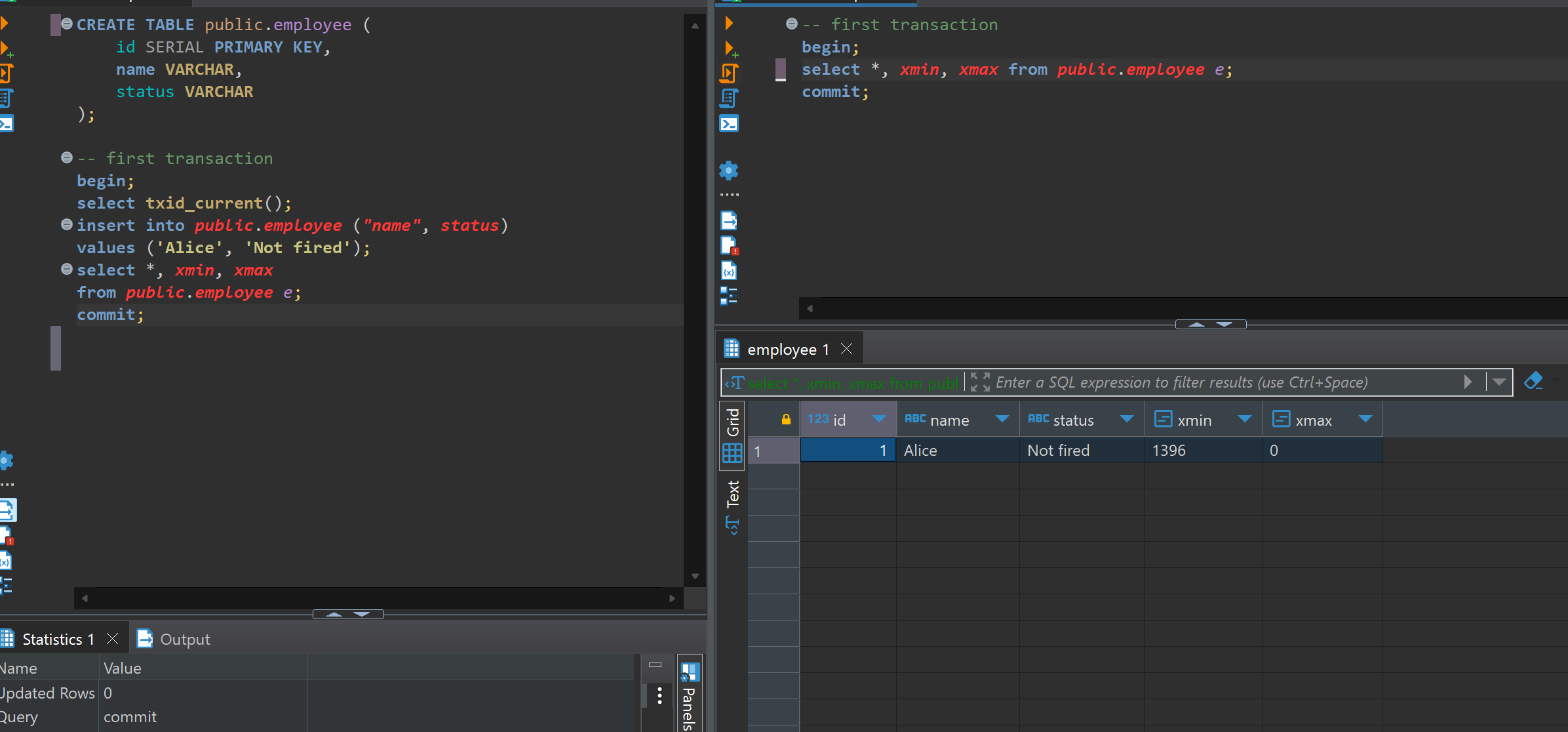
At the beginning, on the right side, it is visible that on the it is empty as inserting has not been done from left side.



After executing insert and select statements on the left side, doing select on the right side does not show anything as on the left side, transaction is not committed.

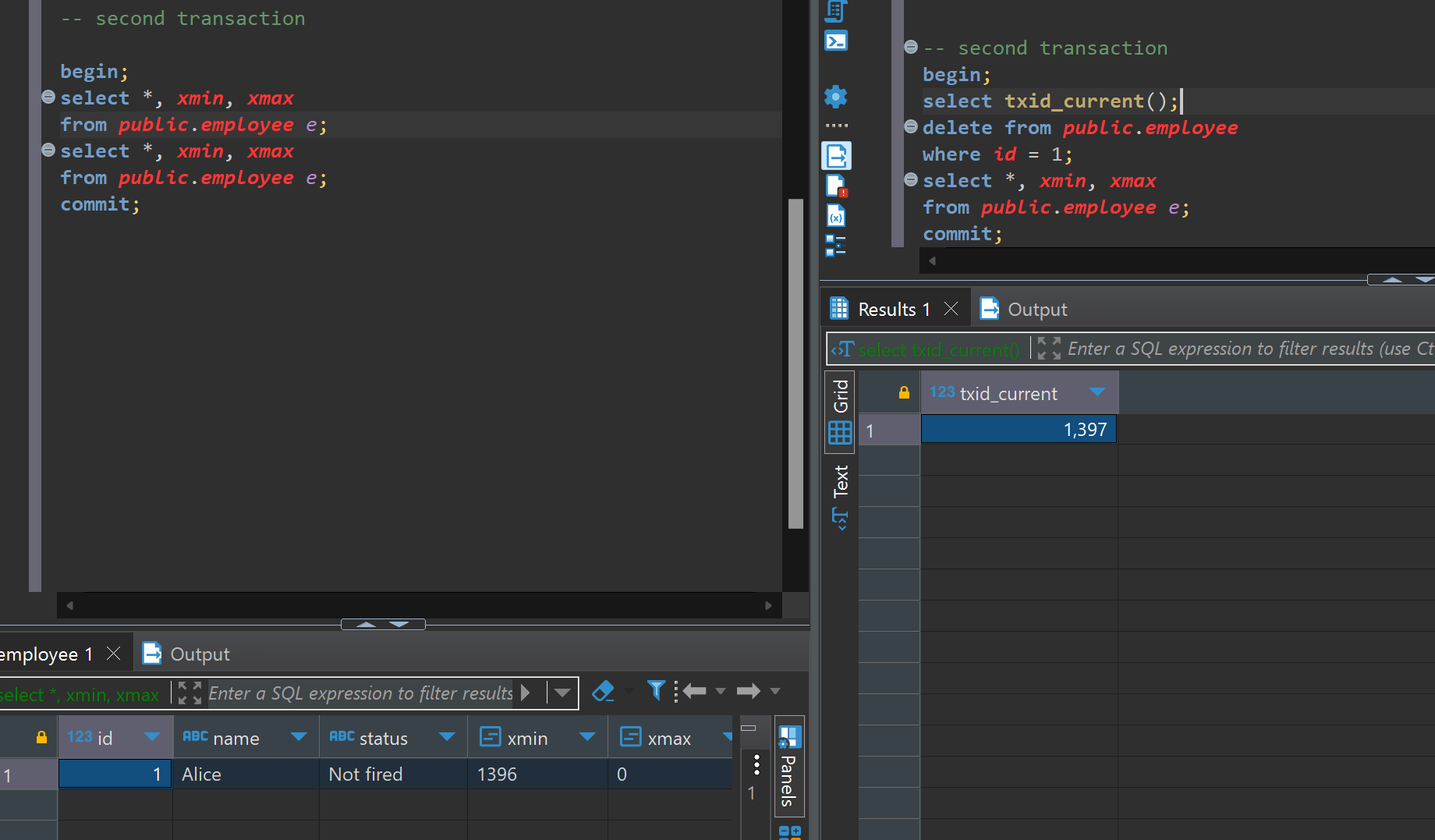


After commiting left side transaction, it is visible for the right side as well

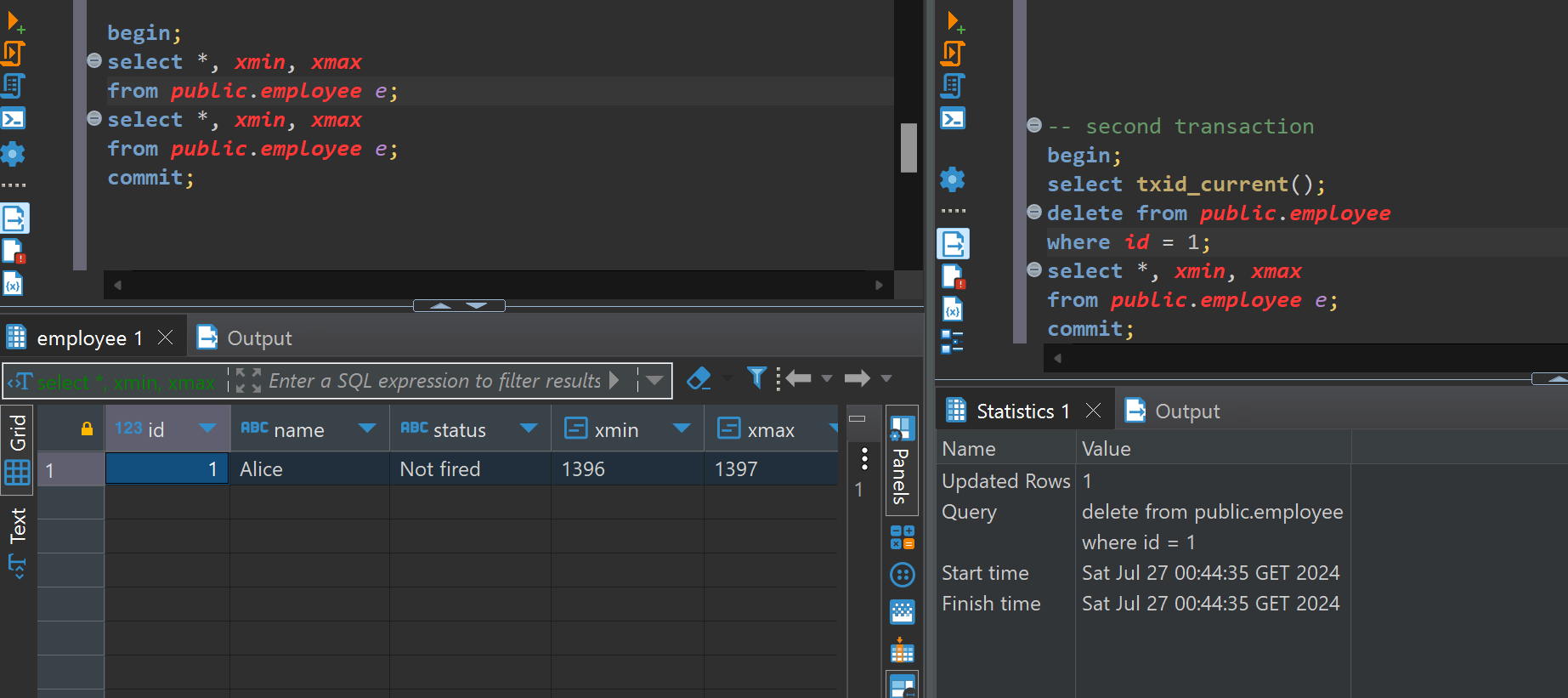


Transaction 2.

Beginning each transaction, and seeing transaction ids on both transactions

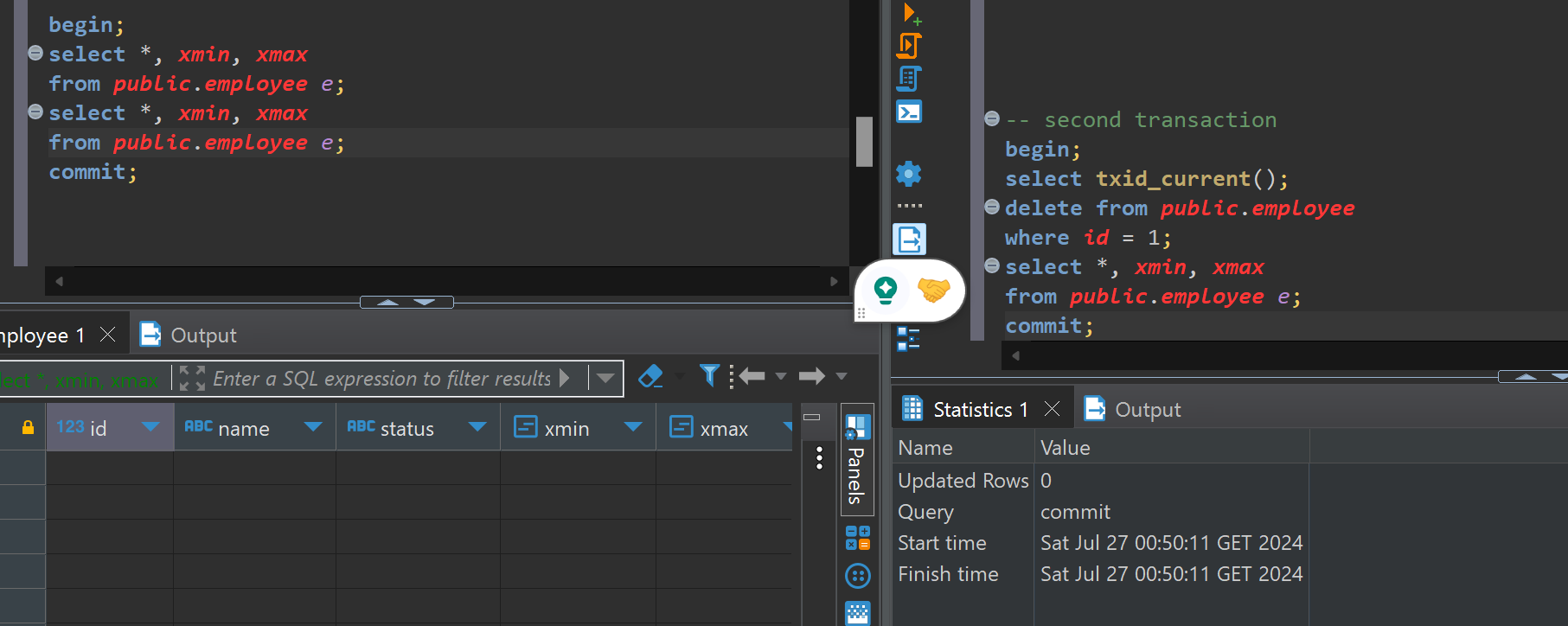


Deleting row from the right side and seeing on the left side, that xmax(deleting id ) is updated to the transaction id from the right side.



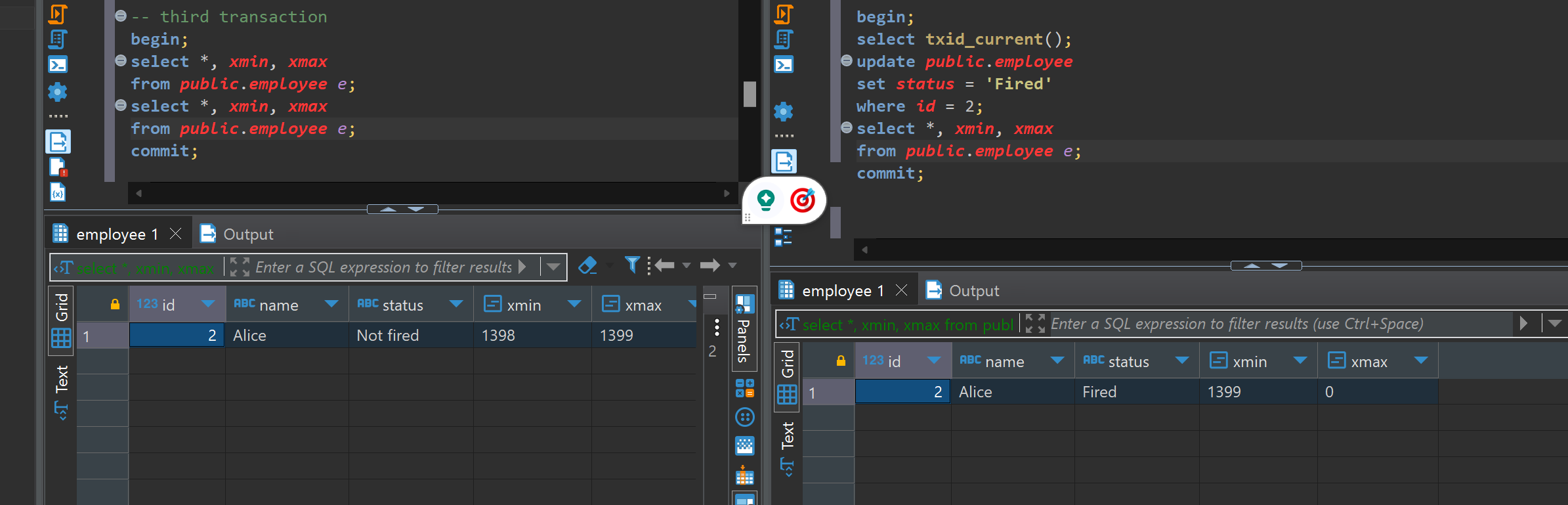
So in the left side, it is still able to see the row, even though it is deleted and xmax shows that.

After commiting right side, the row from the left side is gone.

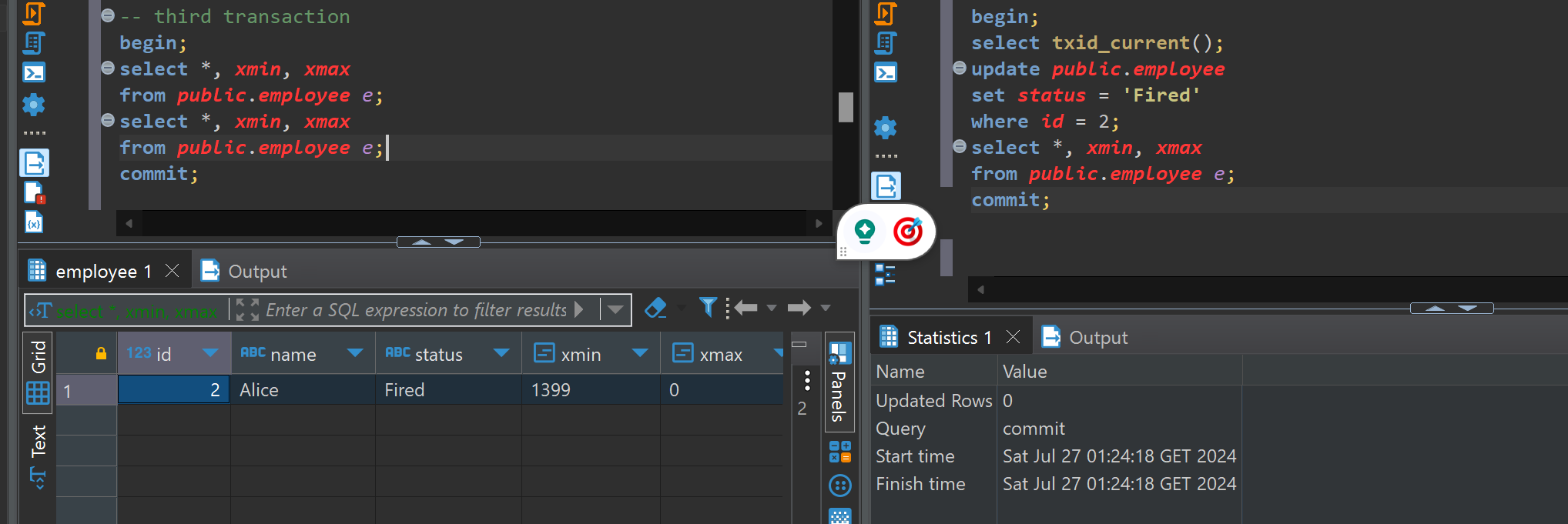


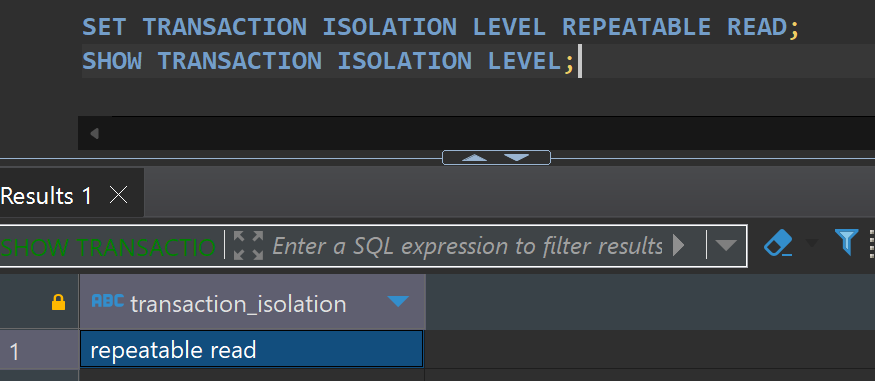
Transaction 3.

In the third transaction, when on the right side I updated the row and it showed the updated version, Alice- Fired, but on the left side, since it is not committed transaction, i can see that xmax is updated to transaction id from the right side, means that this column is deleted, even its updated, it is deleted and rewrited.



After committing, it shows that alice fired updated on the left side as well.



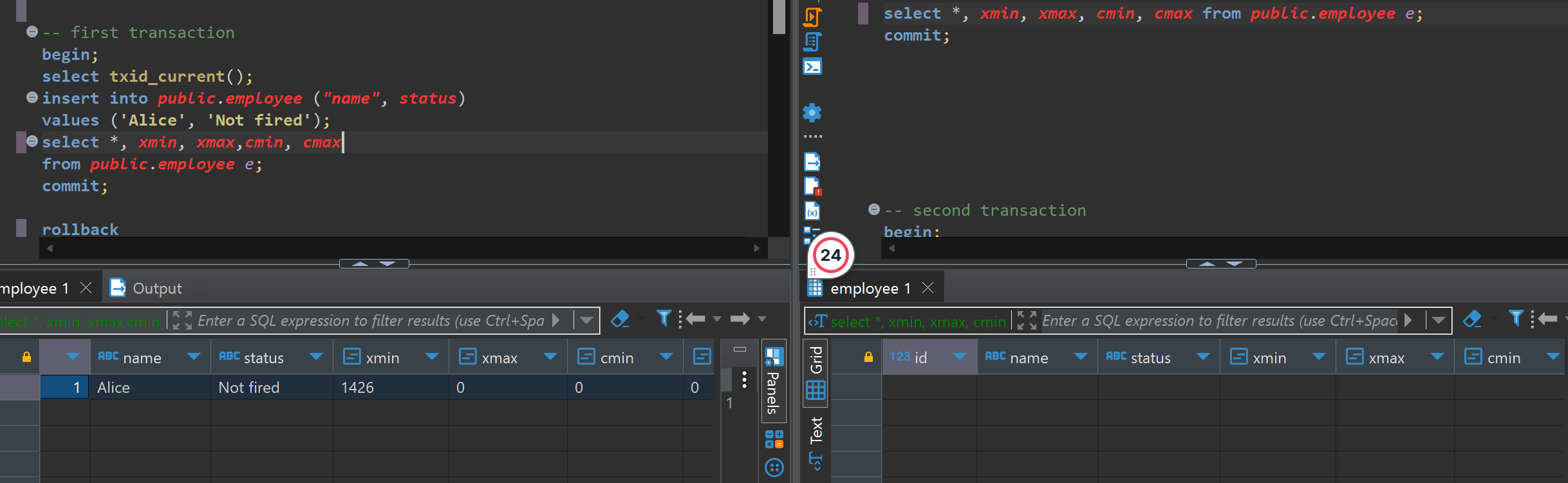
1. 4. 

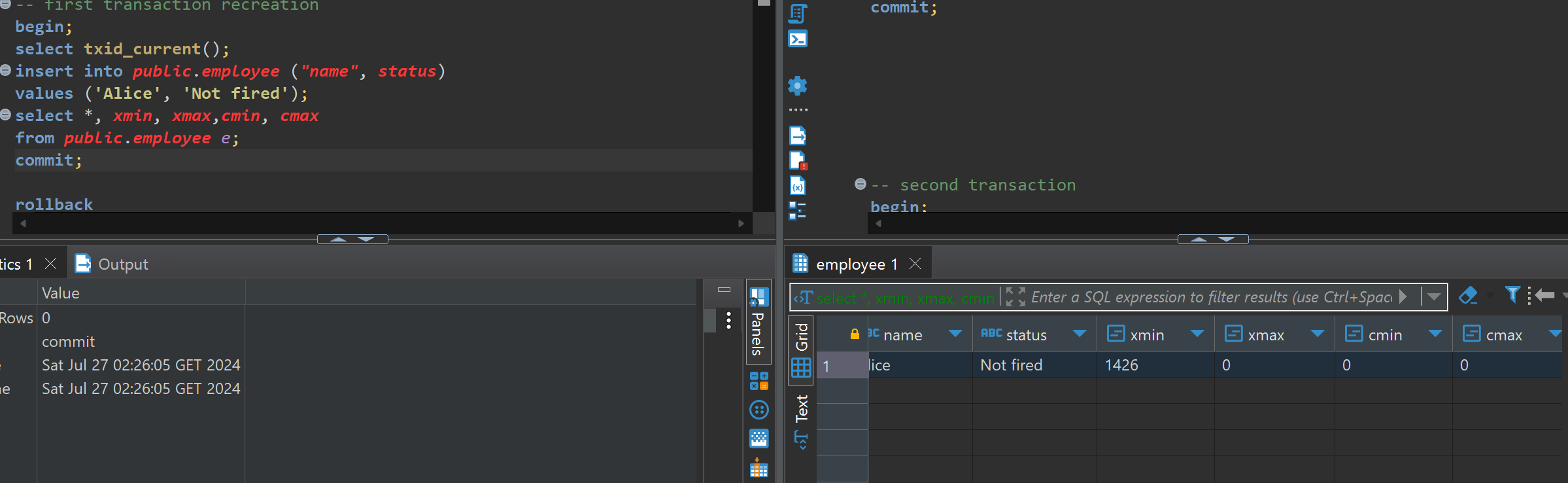
Set isolation level to repeatable read

5.

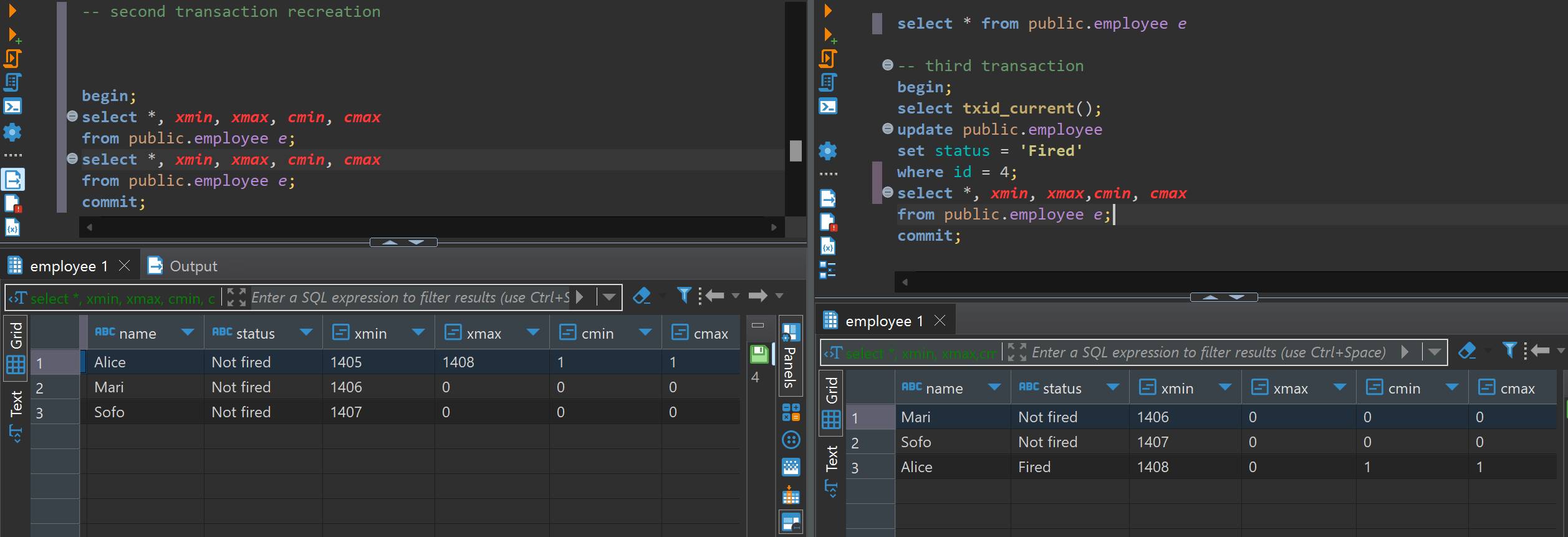
First transaction:

Inserting, repeatable read, i cant see whats happening in the right, cmin, cmax has not changed for the insert on the left side.

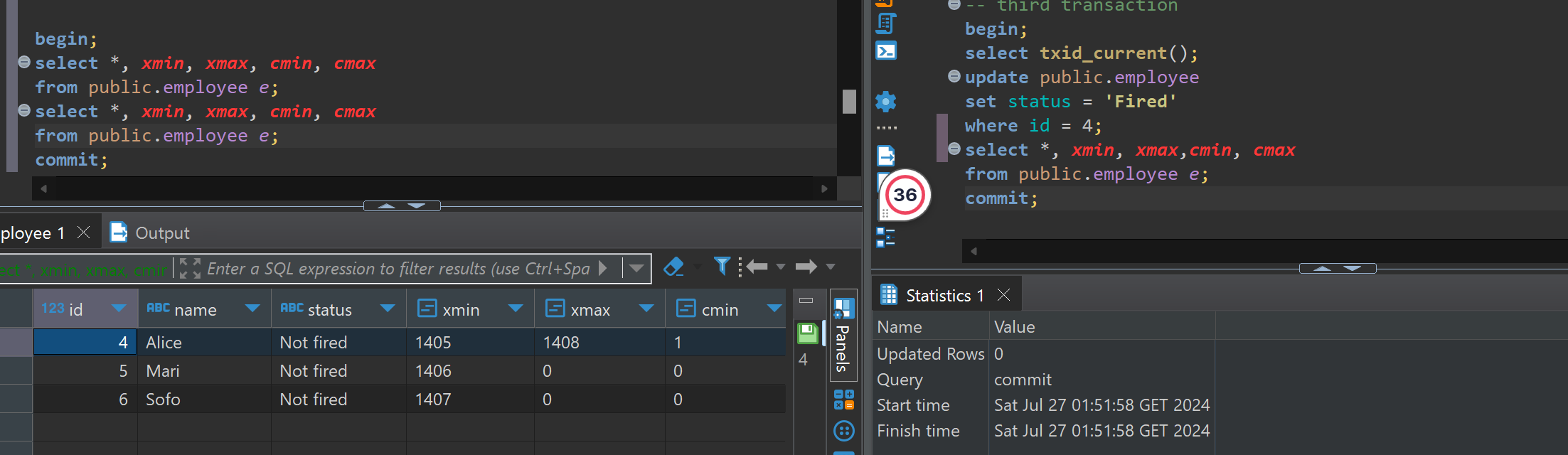


After commiting, 

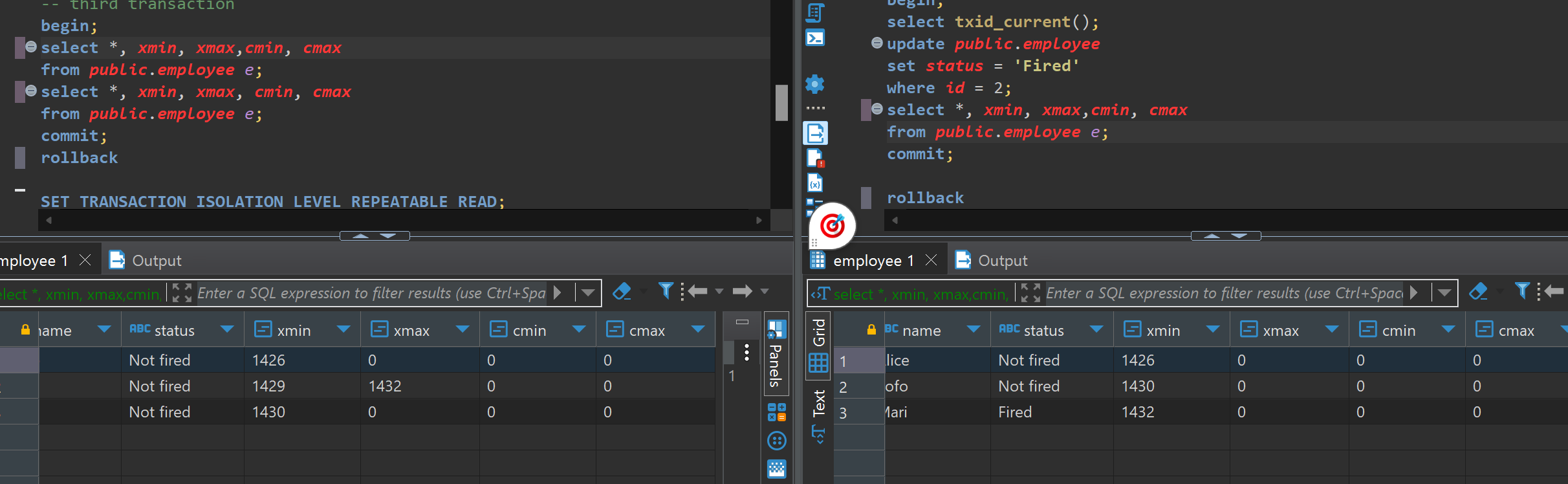
For the second transaction:



Here its visible that cmin and cmax updated to 1, and on the other side even xmax updated with transaction id from right transaction.



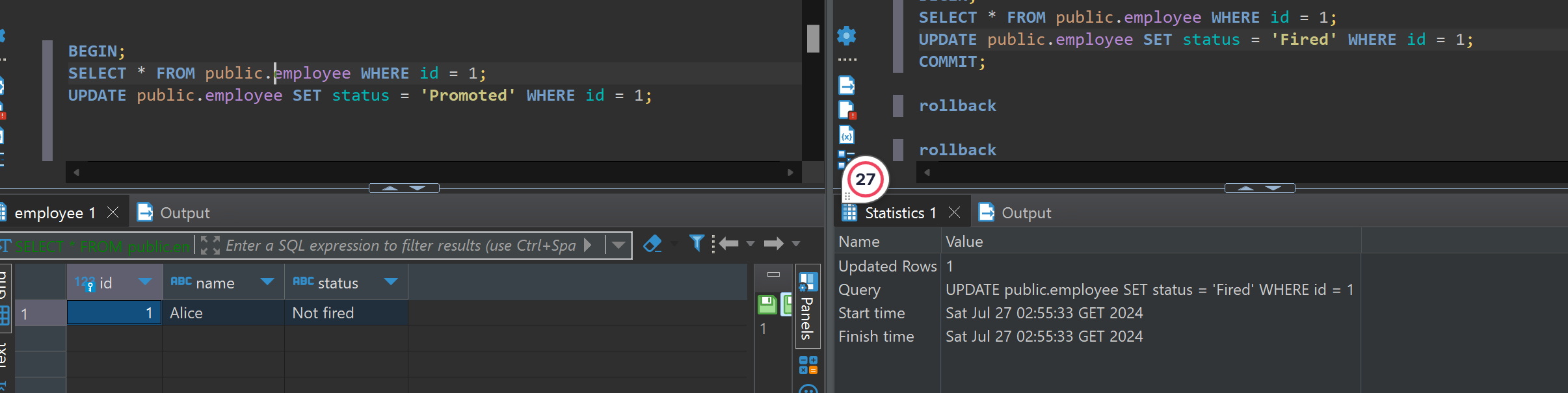
Even though transaction was committed on the right side, row was not updated on the left side.



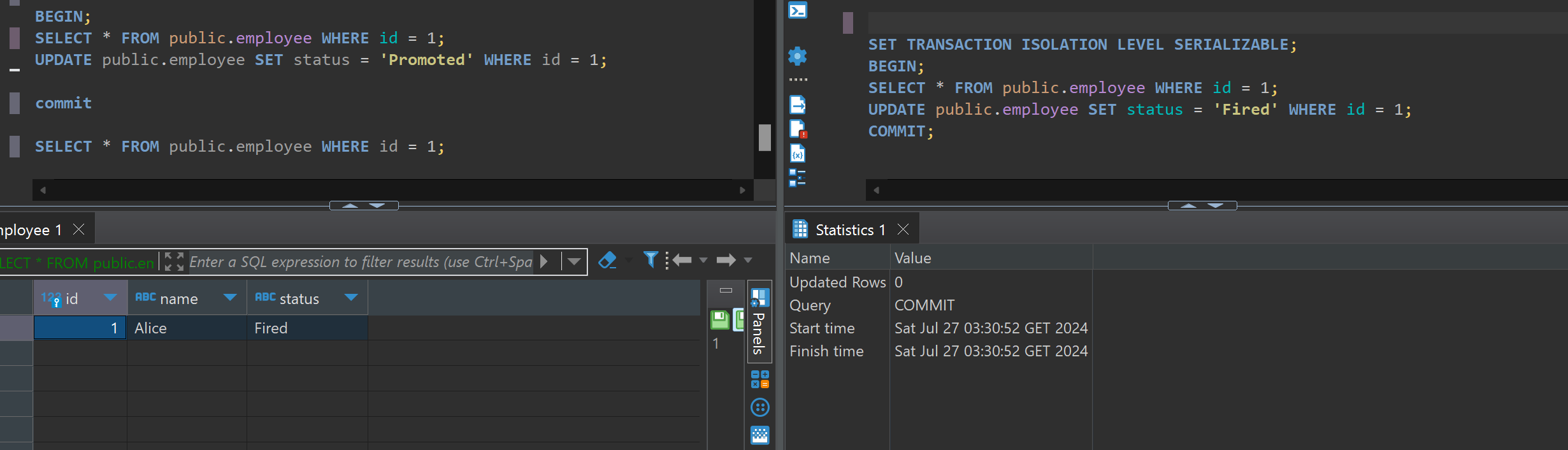
6.

Here, making isolation level serializable, I run update on the right side and updated status to fired, while

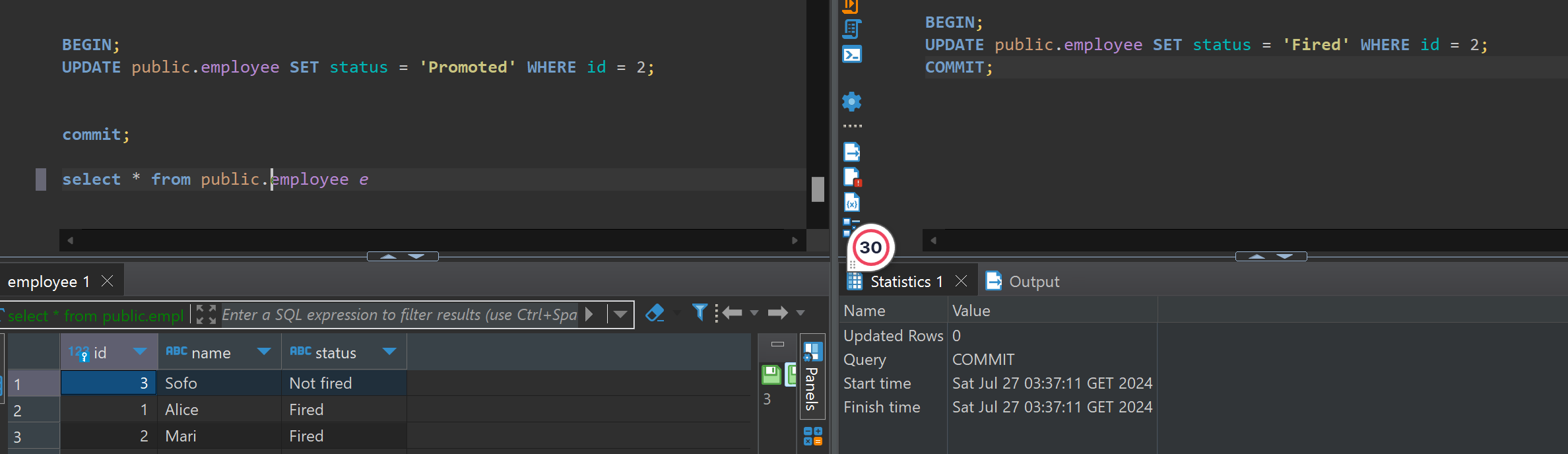
I run select on the left and status is still not fired



After commiting both transactions, update to Promotion was not saved and get lost. Because of the serializable, one option were aborted.



7.



With read committed, it is not prevented to lost update anomalies because it only ensures that each statement sees a consistent snapshot of the database.

concurrent transactions can overwrite each other's changes without awareness. In this case, last one that updated was fired and Mari status was set to Fired.