SQL ACID transaction & stored procedure

Server query optimization and transaction control

About project

In an ACID transaction, if any of these actions fail (for example, if your payment doesn't go through), then the entire transaction fails and none of the changes are made. This ensures that the data in the database remains consistent.

Benefits: ACID transaction is a way to ensure that a group of actions in a database either all happen successfully or none of them happen at all.

Project case study: Let's say you're buying a pair of boots online. When you make the purchase, several things need to happen: the boots need to be added to your cart, your payment needs to be processed, your account needs to be debited the correct amount, the store's account needs to be credited, and the inventory of boots needs to be reduced by one.

- Created a stored procedure routine named **TRANSACTION_ROSE** which will include TCL commands like COMMIT and ROLLBACK, developed the routine based on the given scenario to execute a transaction. **Scenario:** Let's buy Rose a pair of Boots from ShoeShop. So we have to update the Rose balance as well as the ShoeShop balance in the BankAccounts table. Then we also have to update Boots stock in the ShoeShop table. After Boots, let's also attempt to buy Rose a pair of Trainers.
- Created a stored procedure **TRANSACTION_JAMES** to execute a transaction based on the following **scenario**: First buy James 4 pairs of Trainers from ShoeShop. Update his balance as well as the balance of ShoeShop. Also, update the stock of Trainers at ShoeShop. Then attempt to buy James a pair of Brogues from ShoeShop. If any of the UPDATE statements fail, the whole transaction fails the transaction rolls back and commit the transaction only if the whole transaction is successful.

Skills utilised

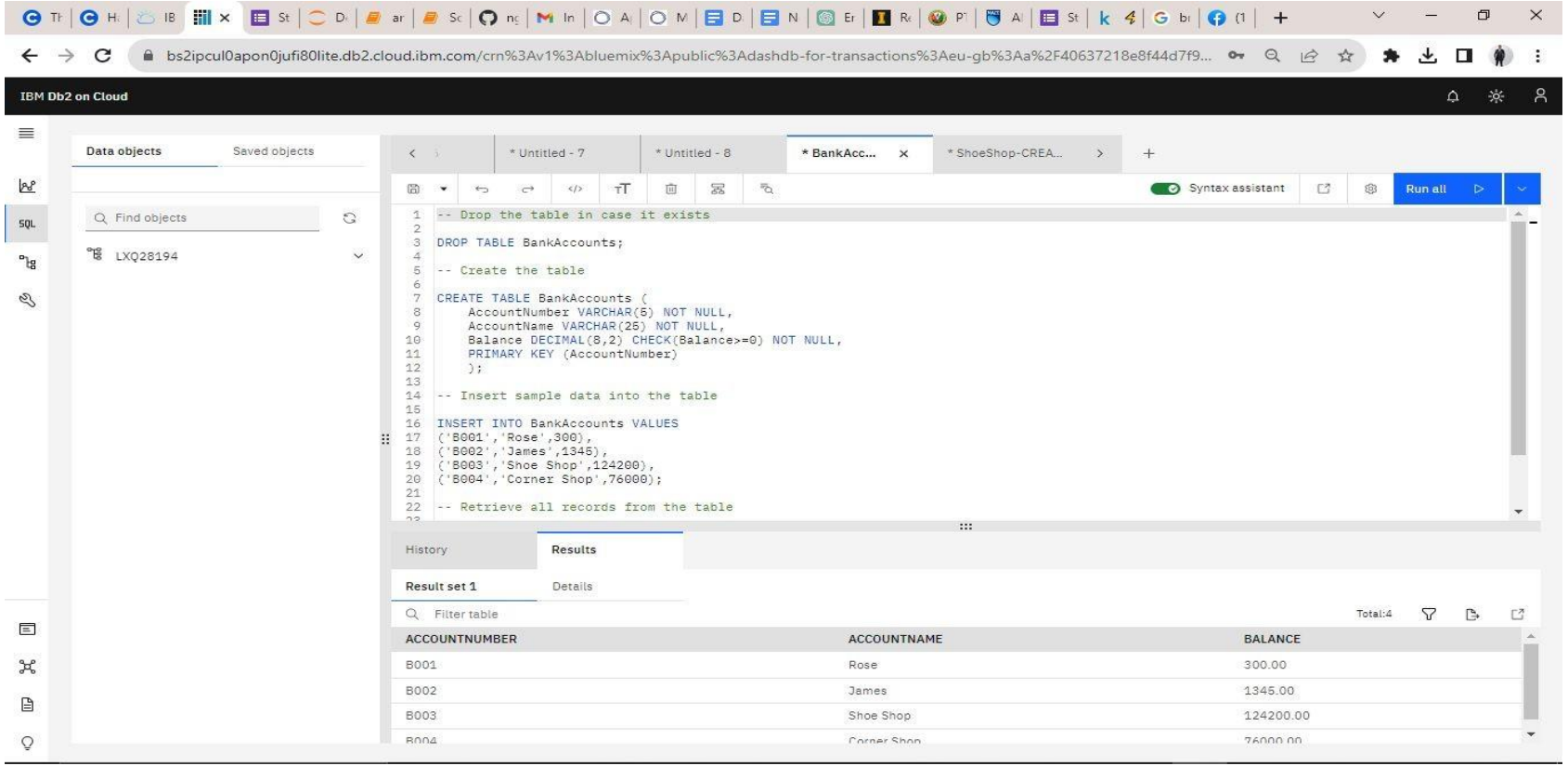
Software used: IBM Db2 cloud database

Project screenshots and explanations

Next eleven slides

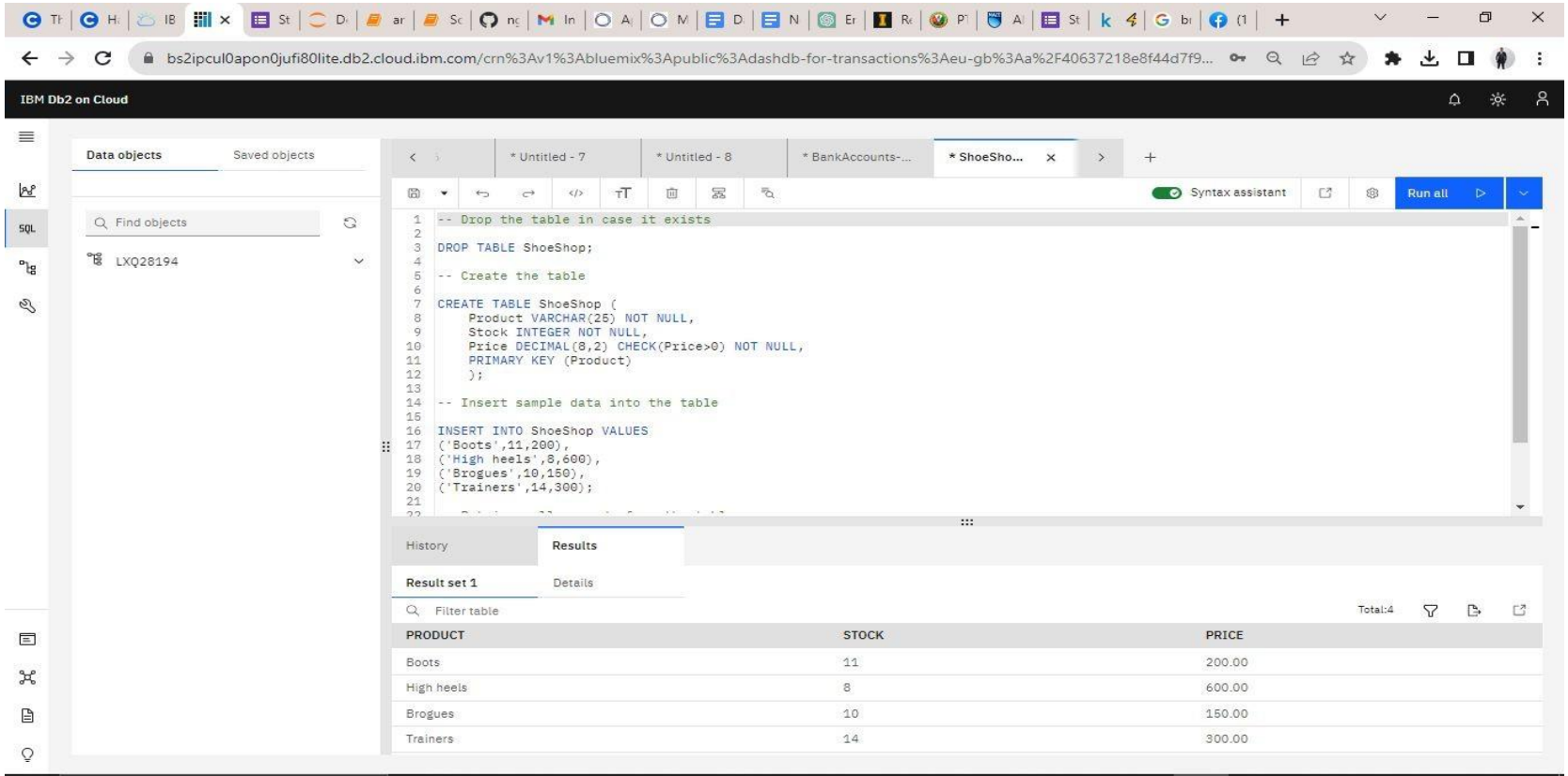
Create database one

This was what the database looks like in IBM Db2 database instance. The table shown contains list of customers account details(including Rose's) with their account balance and account balance.



Create database two

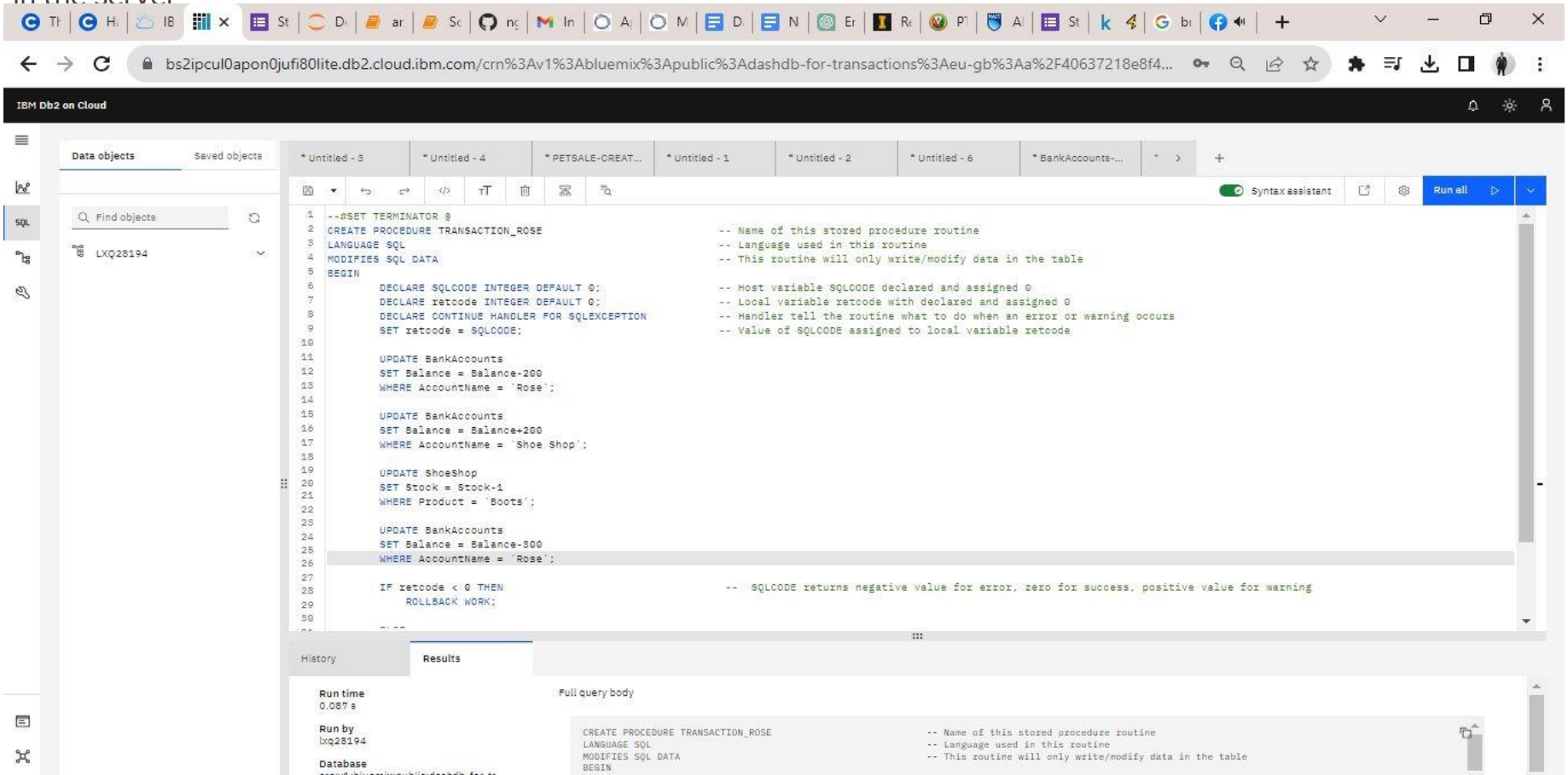
This was what the database looks like in IBM Db2 database instance. The table shown at the bottom of the image contains inventory details of Shoeshop business owner with product name, stock(quantity) and price for each.



TRANSACTION_ROSE STORED PROCEDURE TRANSACTION

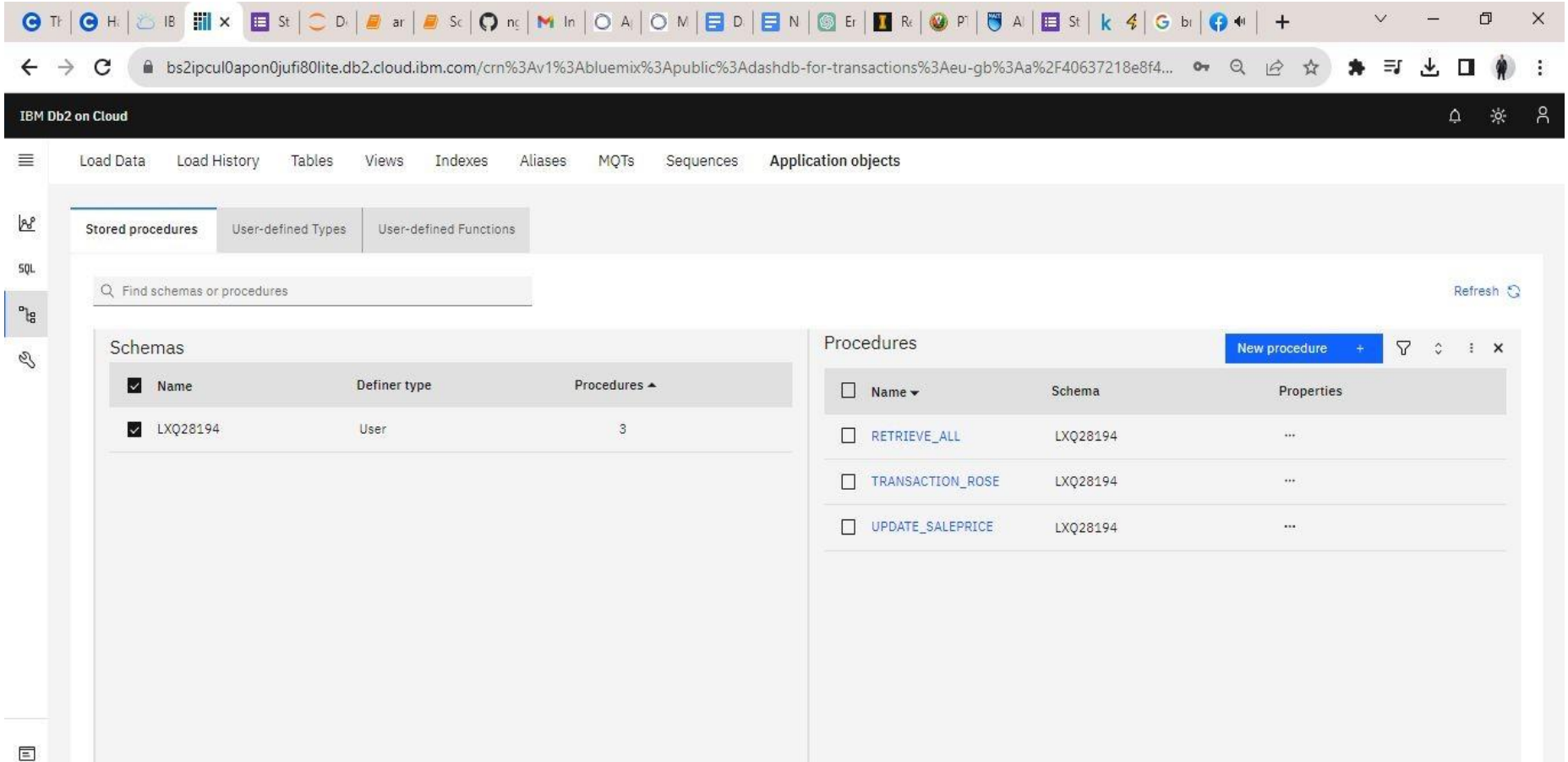
Create Stored Procedure and save in the server

This is screenshot of the stored procedure(TRANSACTION_ROSE) and the SQL successfully created, ran and stored in the server



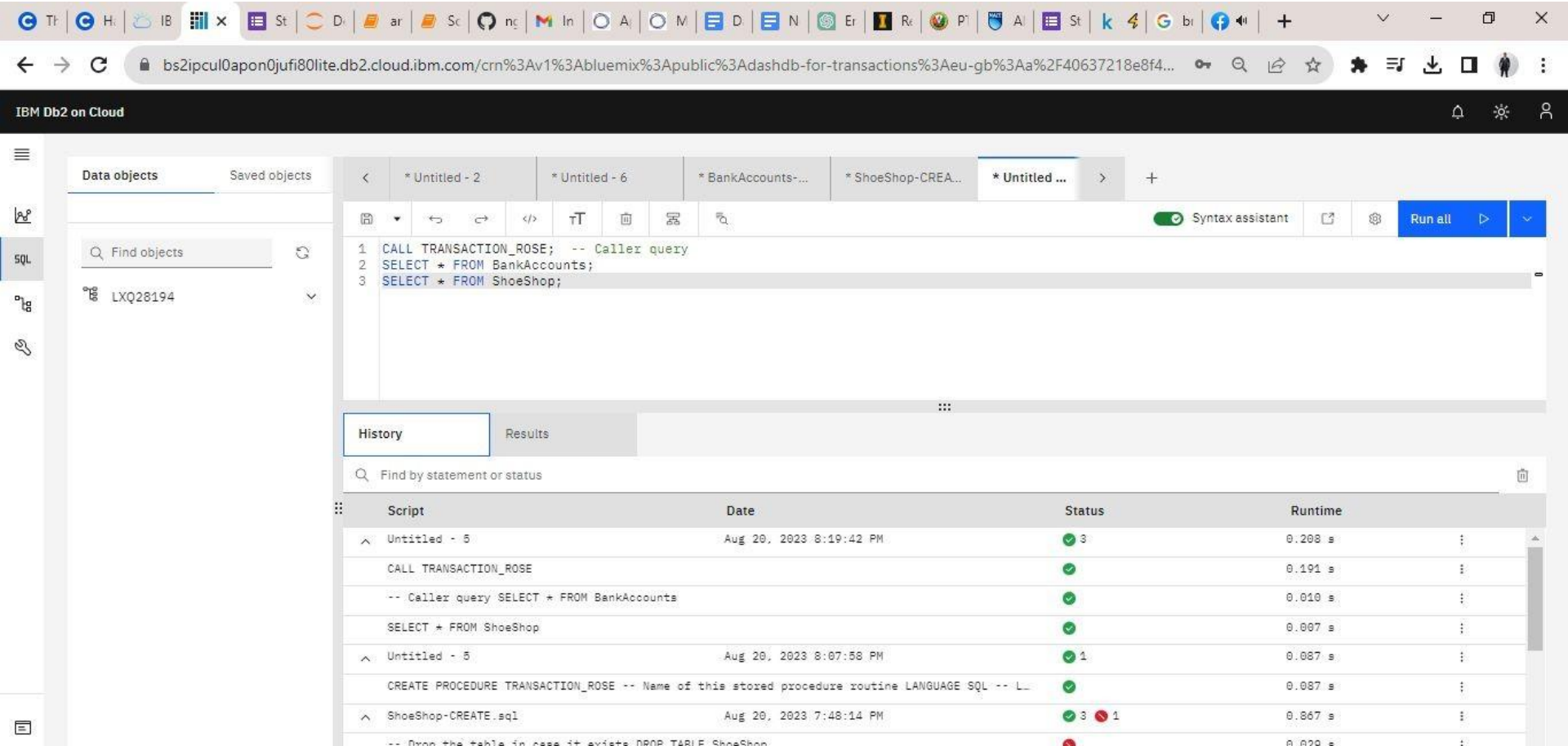
Confirm the stored procedure is saved on the server

This screenshot shows on the right hand side that the **TRANSACTION_ROSE** Procedure now exists in the database server and ready to run when called



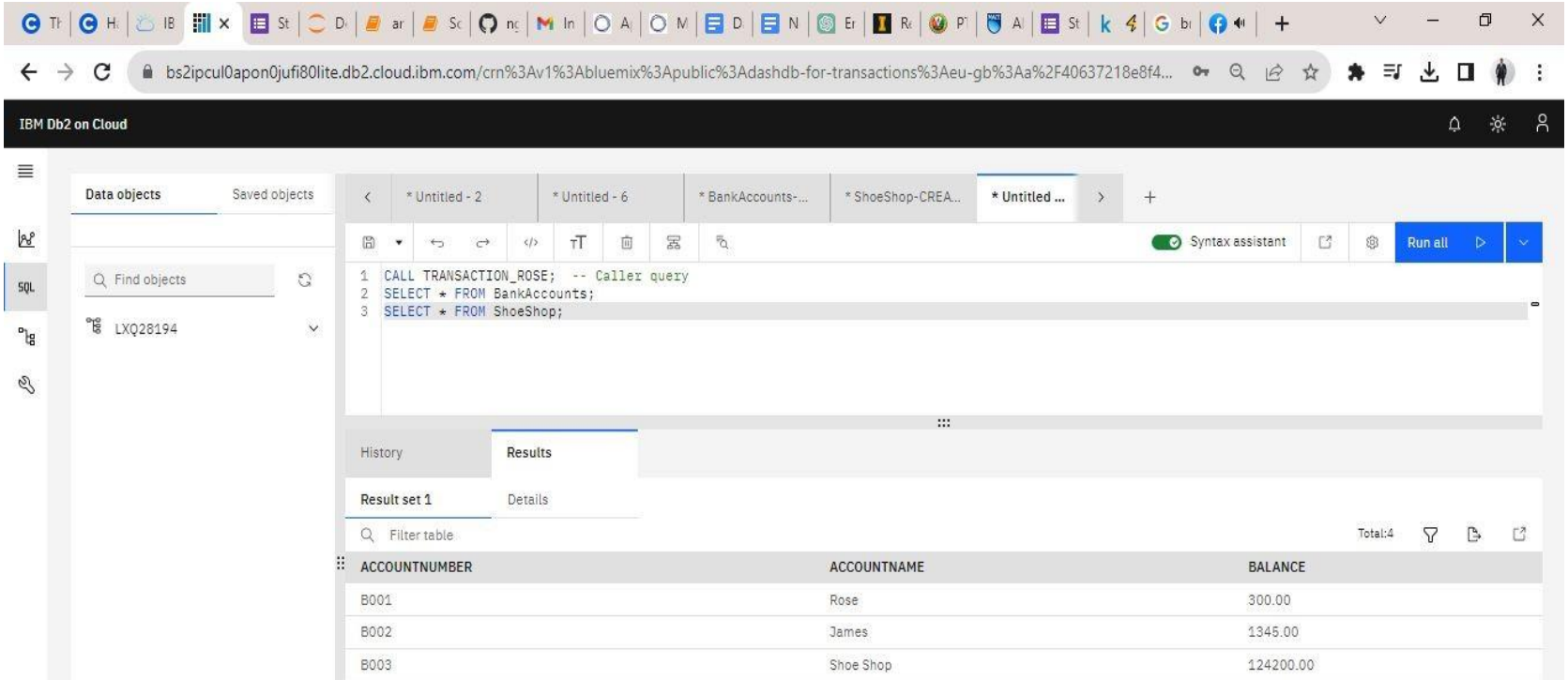
Call stored Procedure

I called the stored procedure **TRANSACTION_ROSE** and it executed successfully, I will need to confirm whether there is a commit action or rollback action.



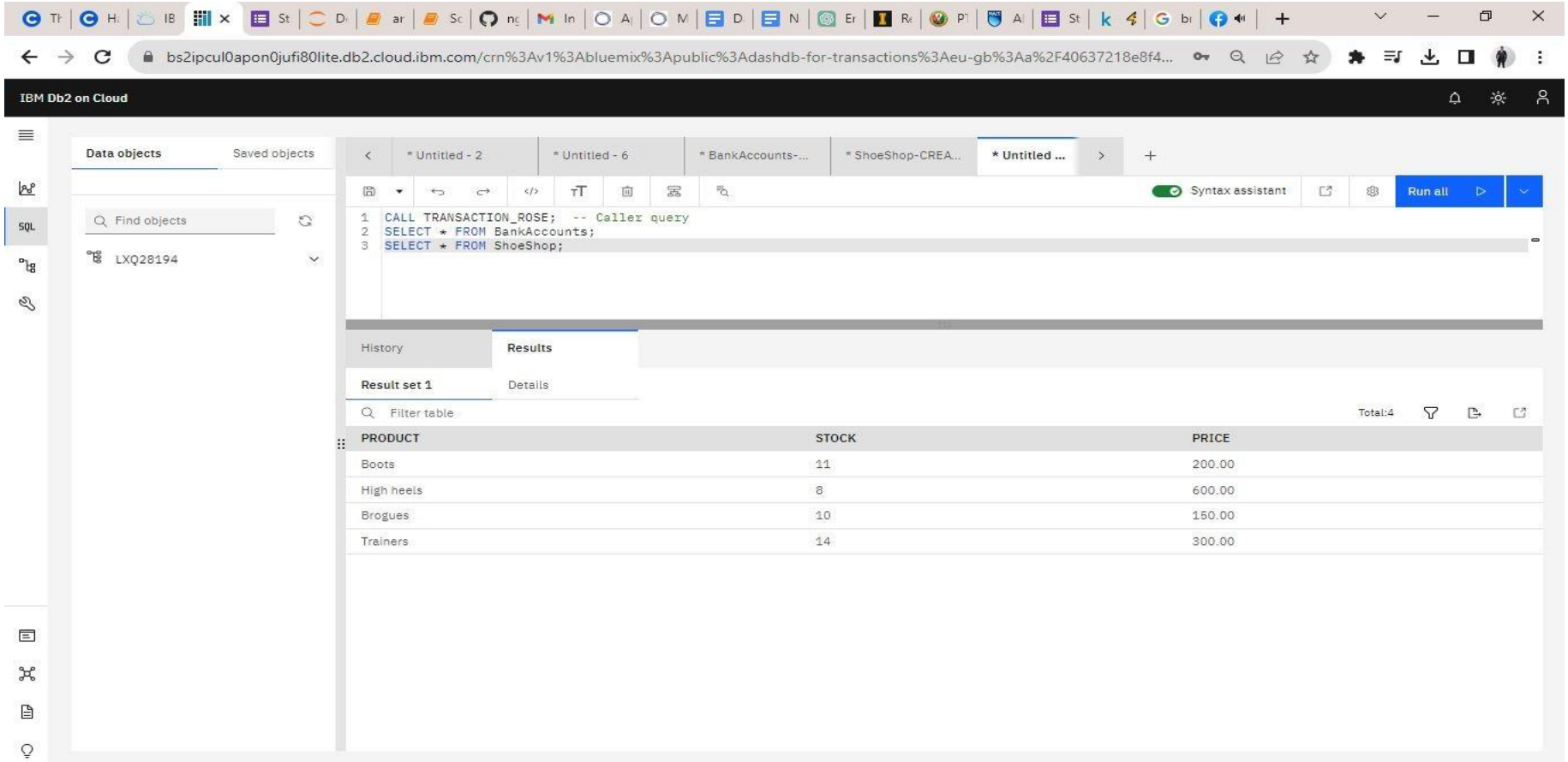
Confirm ROSE Account balance

After the stored procedure call, nothing changed on the account balance of ROSE. **HERE IS WHY**: The first three UPDATEs should run successfully. Both the balance of Rose and ShoeShop should have been updated in the BankAccounts table. The current balance of Rose should stand at 300 - 200 (price of a pair of Boots) = 100. The current balance of ShoeShop should stand at 124200 + 200 = 124400. The stock of Boots should also be updated in the ShoeShop table after the successful purchase for Rose, 11 - 1 = 10. The last UPDATE statement tries to buy Rose a pair of Trainers, **but her balance becomes insufficient** (Current balance of Rose: 100 < Price of Trainers: 300) after buying a pair of Boots. So, the last UPDATE statement fails. **Since the whole transaction fails if any of the SQL statements fail, the transaction won't be committed and the database data records remains unchanged leaving ROSE money untampered with.**



Confirm ShoeShop product inventory details for Boots

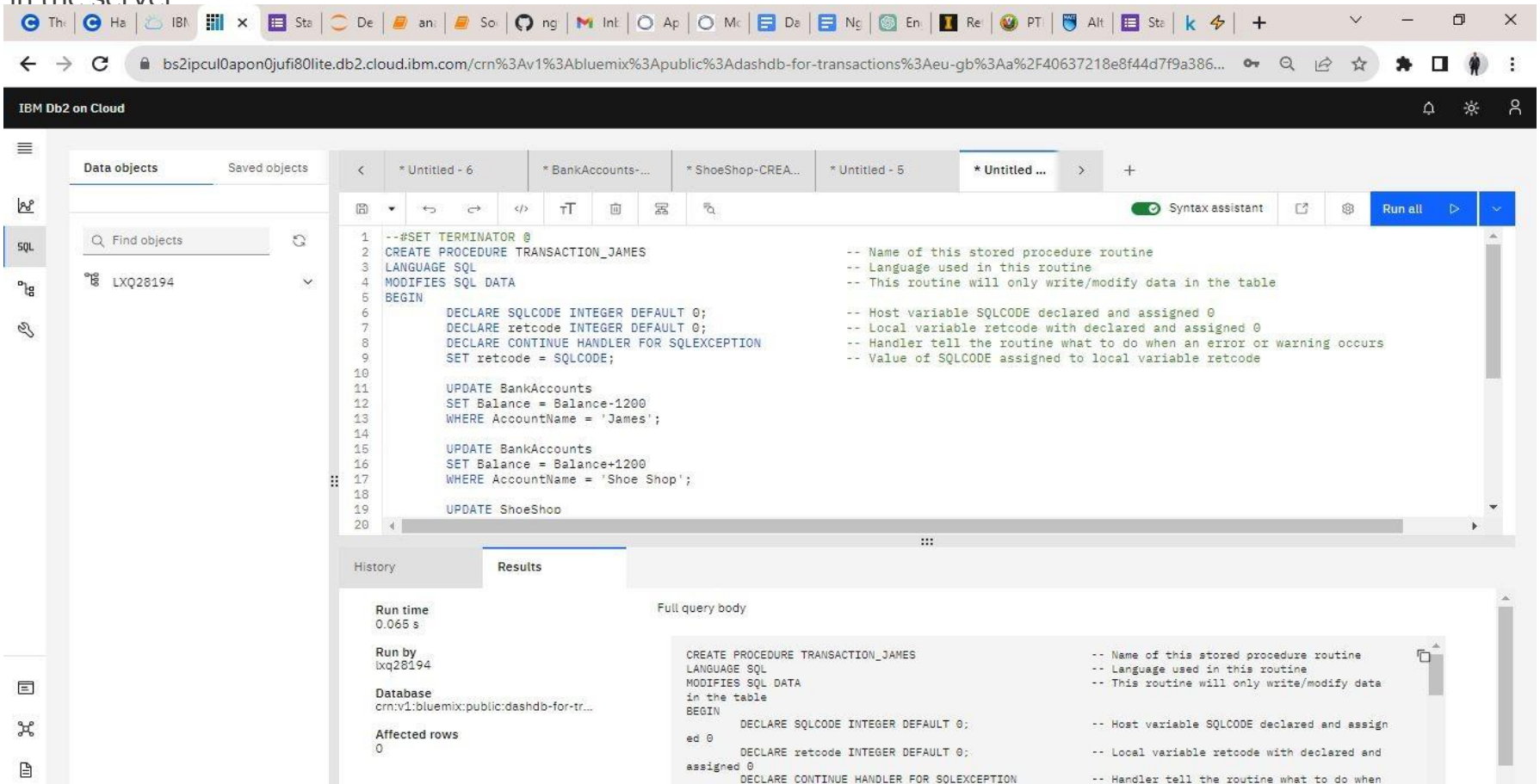
After the **TRANSACTION_ROSE** call, nothing changed on the quantity of **Boots**. **HERE IS WHY**: since one of the **ACID** update transactions failed, all the transaction rolled back and nothing happened to the quantity of **Boots** available in the inventory database which remained at 11. **So the store owner has nothing to worry about!**



TRANSACTION_JAMES STORED PROCEDURE TRANSACTION

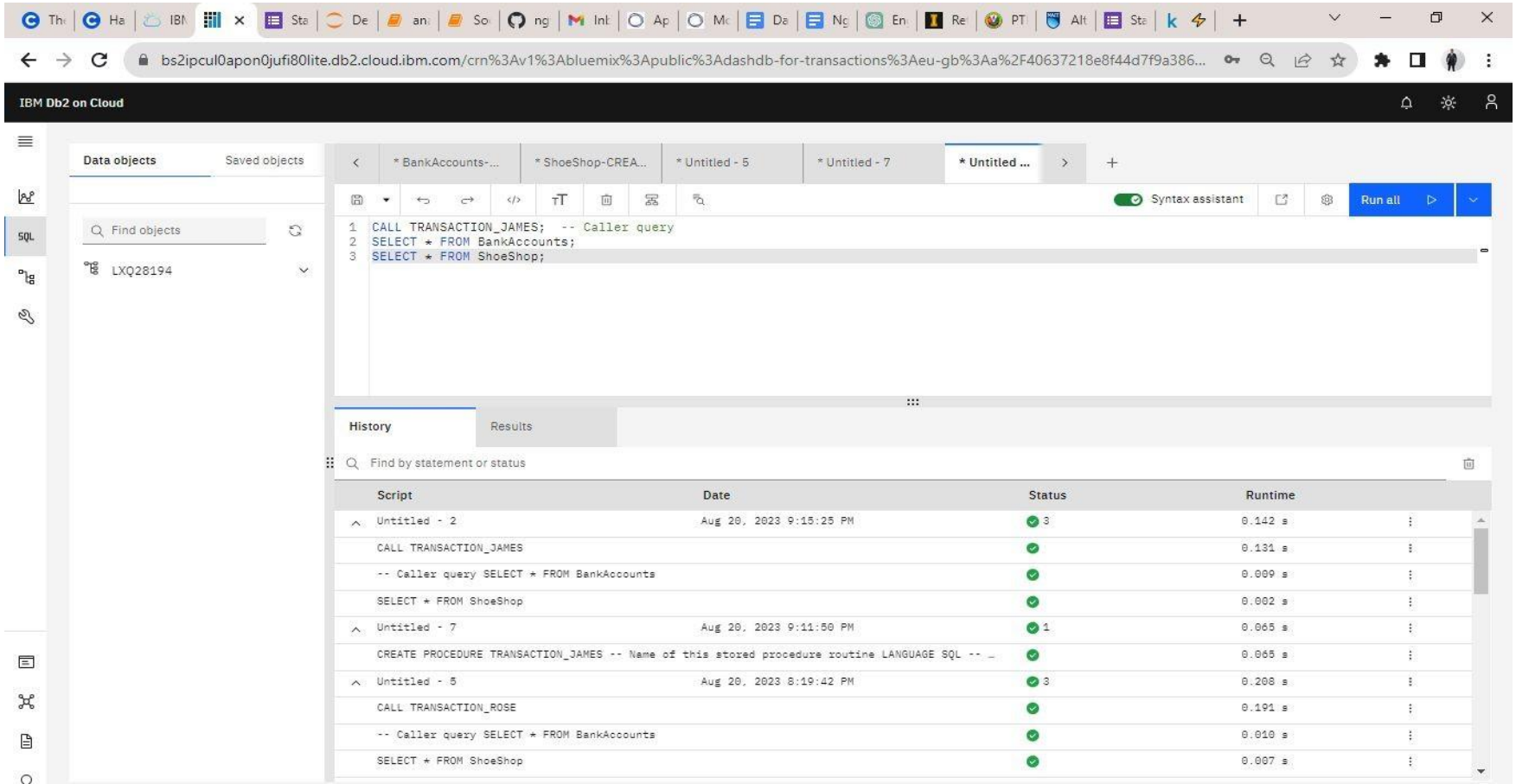
Create Stored Procedure TRANSACTION_JAMES and save in the server

This is screenshot of the stored procedure(TRANSACTION_JAMES) and the SQL successfully created, ran and stored in the server



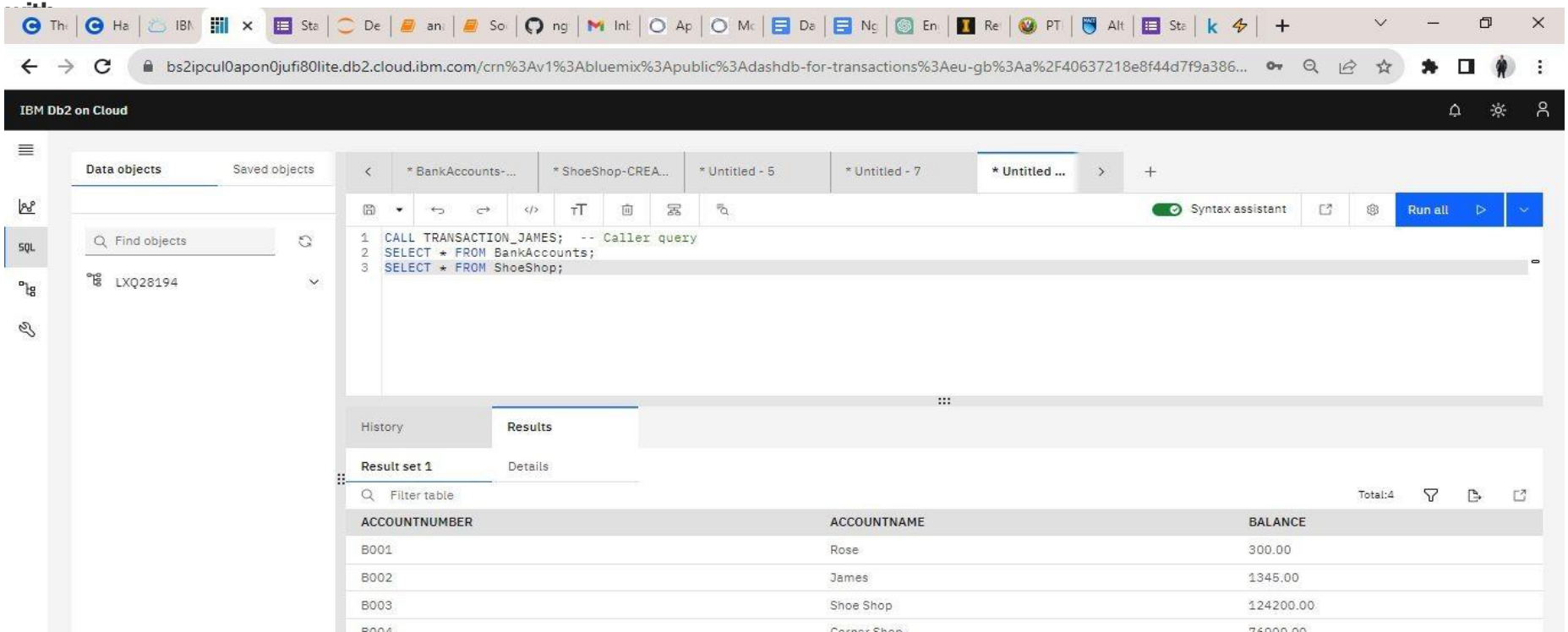
Call stored Procedure

I called the stored procedure **TRANSACTION_JAMES** and it executed successfully, I will need to confirm whether there is a commit action or rollback action.



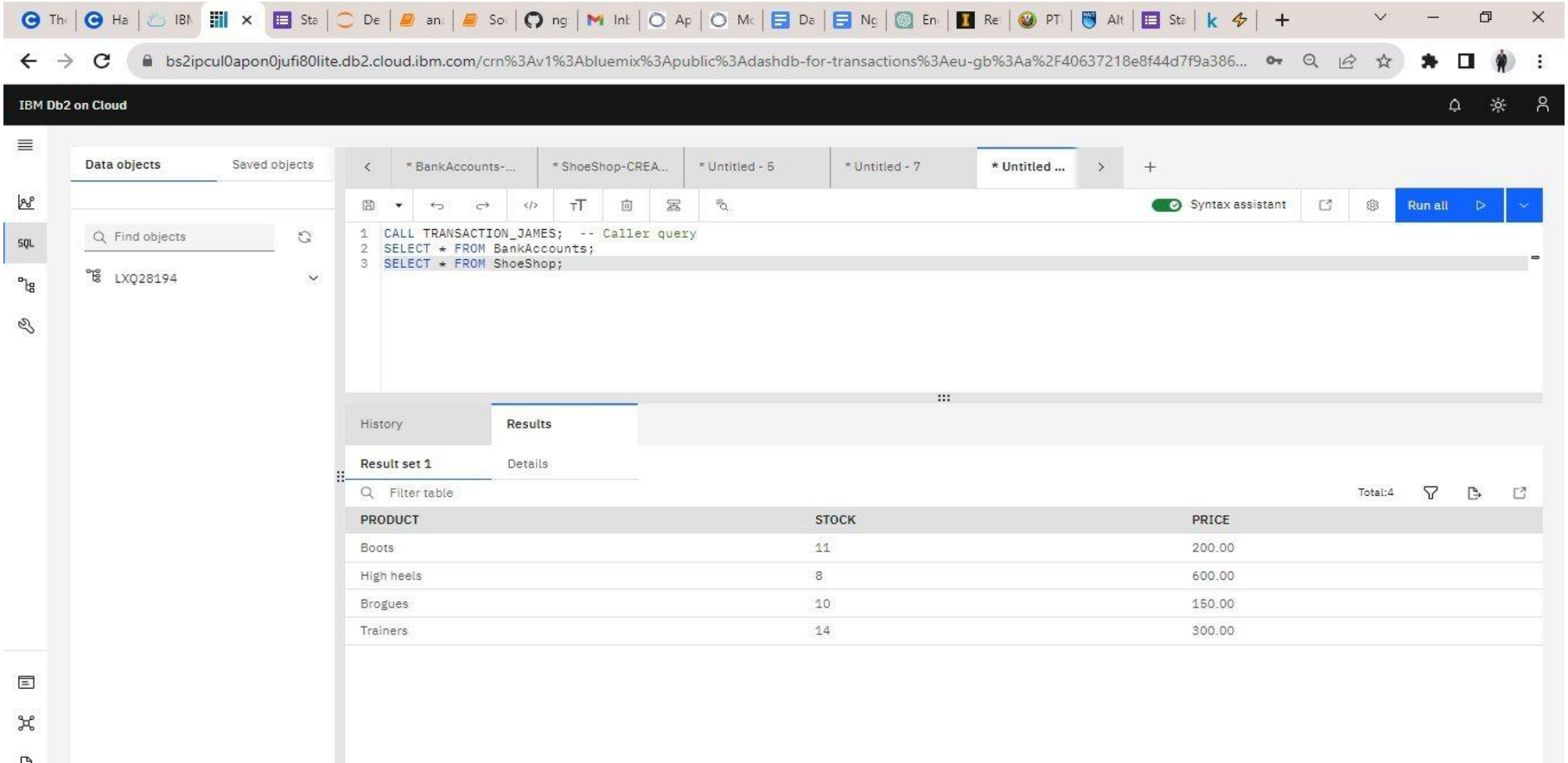
Confirm JAMES Account balance

After the stored procedure call, nothing changed on the account balance of ROSE as shown below. **HERE IS WHY:** The first three UPDATEs should run successfully. Both the balance of james and ShoeShop should have been updated in the BankAccounts table. The current balance of James should stand at 1345 - 1200 (price of 4 Trainers, 300*4) = 1200. The current balance of ShoeShop should stand at 124200 + 1200 = 125400. The stock of Trainers should also be updated in the ShoeShop table after the successful purchase for James, 14 - 4 = 10. The last UPDATE statement tries to buy James a Brogues, **but his balance becomes insufficient** (Current balance of james: 145 < Price of Brogues: 150) after buying a 4 Trainers. So, the last UPDATE statement fails. **Since the whole transaction fails if any of the SQL statements fail, the transaction won't be committed and the database data records remains unchanged leaving JAMES money untampered**



Confirm ShoeShop product inventory details for Trainers and Brogues

After the **TRANSACTION_JAMES** call, nothing changed on the quantity of **Trainer and Brogues**. **HERE IS WHY**: since one of the **ACID** update transactions failed, all the transaction rolled back and nothing happened to the quantity of Trainers/Brogues available in the inventory database which remained at 14 and 10 respectively. **So the store owner has nothing to worry about!**



Conclusion

ACID transactions play a crucial role in maintaining the reliability, integrity, and consistency of data, which directly benefits businesses and their customers by providing a solid foundation for accurate and trustworthy operations.

Contact

Christian Nzeanorue

nzeanoruec@gmail.com

+1(240)337-3535