**Exercise 3: Stored Procedures**

* **Scenario 1:** The bank needs to process monthly interest for all savings accounts.
* **Question:** Write a stored procedure ***ProcessMonthlyInterest*** that calculates and updates the balance of all savings accounts by applying an interest rate of 1% to the current balance.
* The *ProcessMonthlyInterest* procedure updates the balance of savings accounts by adding a 1% interest, setting the LastModified date, and committing the changes.
* It uses a cursor to iterate over all savings accounts and applies the interest rate to each account's balance.
* If an error occurs during execution, the transaction is rolled back, and an error message is printed.
* After defining and executing the procedure, a query is run to verify the updated data in the *Accounts* table for savings accounts.

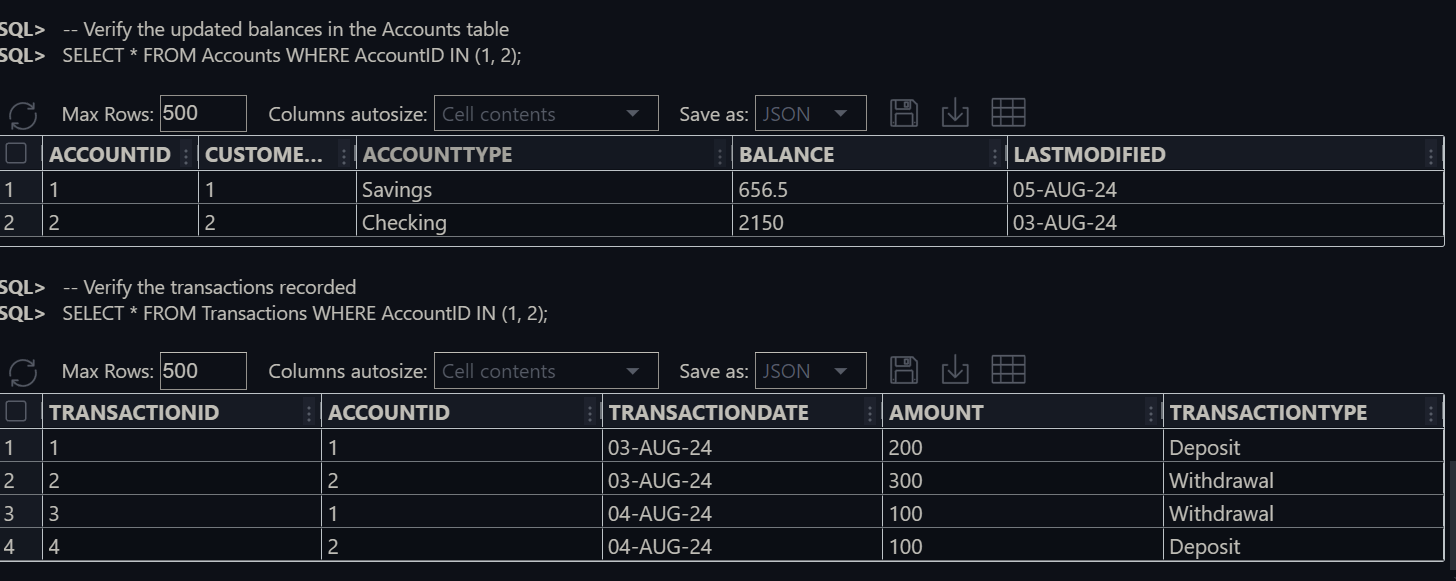
**Here is the Output:**

* **Scenario 2:** The bank wants to implement a bonus scheme for employees based on their performance.
* **Question:** Write a stored procedure **UpdateEmployeeBonus** that updates the salary of employees in a given department by adding a bonus percentage passed as a parameter.
* The *UpdateEmployeeBonus* procedure updates employee salaries by adding a percentage-based bonus for a specified department and commits the changes.
* It outputs the number of updated rows and handles errors by rolling back changes and displaying an error message.
* A test block calls the procedure to update bonuses for the IT department by 10%, and also handles any exceptions during the test.
* A *SELECT* statement retrieves and displays the updated salaries for the IT department to verify the changes.

****

**Here is the Output:**

* **Scenario 3:** Customers should be able to transfer funds between their accounts.
* **Question:** Write a stored procedure ***TransferFunds*** that transfers a specified amount from one account to another, checking that the source account has sufficient balance before making the transfer.
* **Procedure Creation**: Defines a stored procedure *TransferFunds* to transfer a specified amount between two accounts, checking for sufficient balance before updating account balances and recording the transaction.
* **Transaction Management**: Includes error handling with exceptions to rollback changes and print messages for insufficient funds or other errors, ensuring database integrity.
* **Sample Data and Sequence**: Inserts sample account records into the *Accounts* table and creates a sequence *TransactionID\_SEQ* for generating unique transaction IDs.
* **Testing and Verification**: Tests the *TransferFunds* procedure with both sufficient and insufficient balances, and then verifies the updated balances and transactions in the *Accounts* and *Transactions* tables.

**Here is the output:**