1. მოვნახოთ ყველა კლიენტი, ვისაც გააჩნია როგორც სესხი ასევე დეპოზიტი. არ გამოვიყენოთ DISTINCT და JOIN
2. სესხებზე და დეპოზიტებზე გავაკეთოთ ოთხივე JOIN-ი და ავხსნათ რა გასხვავებულ შედეგს მოგვცემს თითოეული.
3. ავაწყოთ კომპლექსური ჯოინები, 4-5 ცხრილის დაკავშირებოთ (ცხრილებს მივანიჭოთ Alias-ები).
4. გავაკეთოთ არავალიდური ჯოინები და ავხსნათ რა რეზულტატს მივიღებთ. უტოლობის დაწერით ან მეტობით ან ნაკლებობით. სიტყვიერად აღწერეთ რას გამოიწვევს.
5. DB_BANK ბაზაზე ავაწყოთ ყველა შესაძლო PK-FK კავშირები და Screen-ების სახით გამომიგზანეთ. სადაც ნათლად გამოჩნდება რომელია Primary Key და Foreign Key ცხრილები და ასევე PK და FK სვეტი.
6. ჩამოვწეროთ თითეულ ცხრილს შორის კავშირი ტიპი: წყვილები
   Customers >> Accounts
   Customers >> Loans
   Loans >> LoanAccouns
   Customer >> Deposits
   Accounts >> Overdrafts
   Transactions >> Accounts
   TransactionTypes >> Transactions
   უნდა დავწეროთ ჩვენს მიერ შემუშავებული ფრაზებით: თითეული მომხარებელი დაკავშირებულია მრავალ შეკვეთასთან თითეული შეკვეთა ერთ მომხმარებელთან და ა.შ
7. დავადგინოთ თითეული ტრანსაქციაში debit-ის და credit-ის ანგარიში ვის ეკუთვნის , ანუ რეზულტატში თითეულ ტრანზაქციას უნდა მიედგას ორ-ორი სახელი და გვარი, ერთი დებიტის ანგარიშის მფლობელის, მეორე - კრედიტის.
8. თუ კი მეოთხე დავალება სწორად ამოხსენით, ნახავთ, რომ დეპოზიტიც და სესხიც სულ გააჩნია 40 მომხმარებელს, მაგრამ თუ გავუშვებთ ამ query-ს SELECT * FROM loan.Loans as l JOIN Deposits as d ON l.CustomerID = d.CustomerID WHERE l.CustomerID = 115 ვნახავთ რომ და-Select-დება 18 ჩანაწერი, თუ გვინდა join-ები ნორმალურად გვესმოდეს, თითეულმა თქვეთაგანმა სიტყვიერად უნდა დაწეროს დავალებაში რატომაა 18.
9. JOIN-ის და DISTINCT-ის გამოყენების გარეშე ვაჩვენოთ ისეთი მომხამრებლები, რომლებსაც მხოლოდ ან სესხი აქვს ან დეპოზიტი, ორივე ერთად არა.

P.S. Transactions ცხრილში DebitAccountID არის გადამრიცხავი ანგარიშის ID და CreditAccountID არის მიმღების ანგარიშის ID.

ასევე ერთ-ერთ კავშირზე როგორც მახსოვს ხარვეზია, თუ არ ვცდები Transactions და TransactionTypes-ის დროს, TinyINT გადააკეთეთ INT-ად და გასწორდება.