Baza danych

System Finansowy

System transakcyjny składa się z 4 tabel powiązanych ze sobą relacjami na schemacie powyżej. Tabela "Customers" zawiera prywatne dane klientów, tabela "Accounts" dane dla kont klientów, tabela "Money_transfer" gromadzi transakcje wykonane przez klientów, tabela "Types" to dostępne typy transakcji w systemie. Tabela "Accounts" zawiera pole "ACC_STATUS", przechowujące informację o statusie danego konta. Tłumaczenie: 1 - OPENED, 2 - CLOSED, 3 - BLOCKED. Pole "ACC_TYPE" przechowuje informację o typie danego konta. Tłumaczenie: 1 - BUY, 2 - SELL. Aby klient mógł zrealizować transakcję ze swojego konta, wymagane jest, aby konto było otwarte w momencie wykonania transakcji, czyli w dniu TRN_DATE.

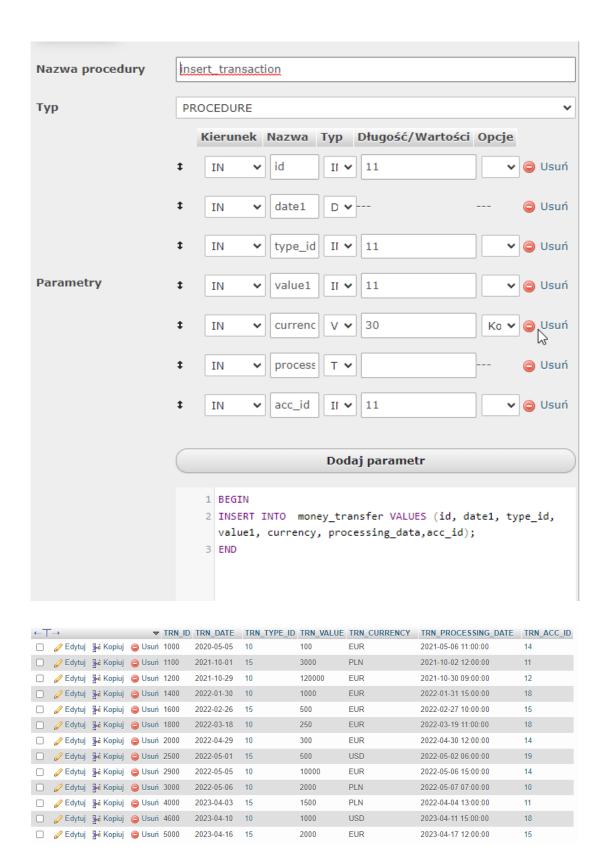
- 1. Stworzyć bazę danych. Utworzyć wszystkie tabele i relacje zgodnie ze schematem. Załadować dane do 3 tabel.
- 2. Napisać procedurę "insert_transaction", w taki sposób, aby poniższe wywołania były możliwe. Załadować dane do tabeli "Money_transfer".

```
CALL insert_transaction(1000, STR_TO_DATE('05-05-2020', '%d-%m-%Y'), 10, 100, 'EUR', '2021-05-06 11:00:00', 14);
```

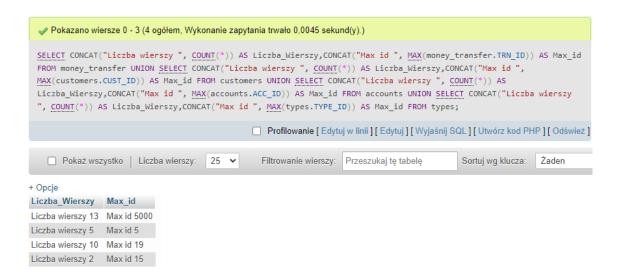
CALL insert_transaction(1200, STR_TO_DATE('29-10-2021', '%d-%m-%Y'), 10, 120000, 'EUR', '2021-10-30 09:00:00', 12);

CALL insert_transaction(1400, STR_TO_DATE('30-01-2022', '%d-%m-%Y'), 10, 1000, 'EUR', '2022-01-31 15:00:00', 18);

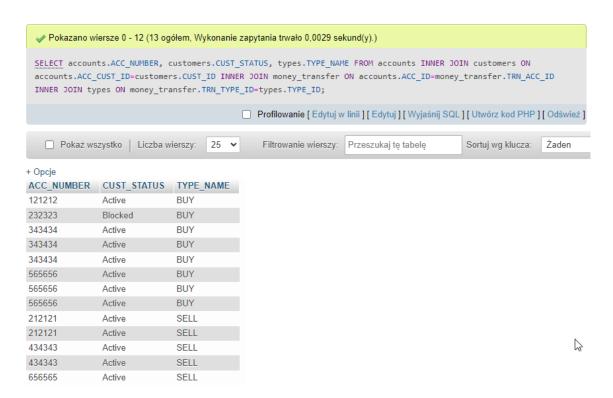
CALL insert_transaction(1600, STR_TO_DATE('26-02-2022', '%d-%m-%Y'), 15, 500, 'EUR', '2022-02-27 10:00:00', 15);



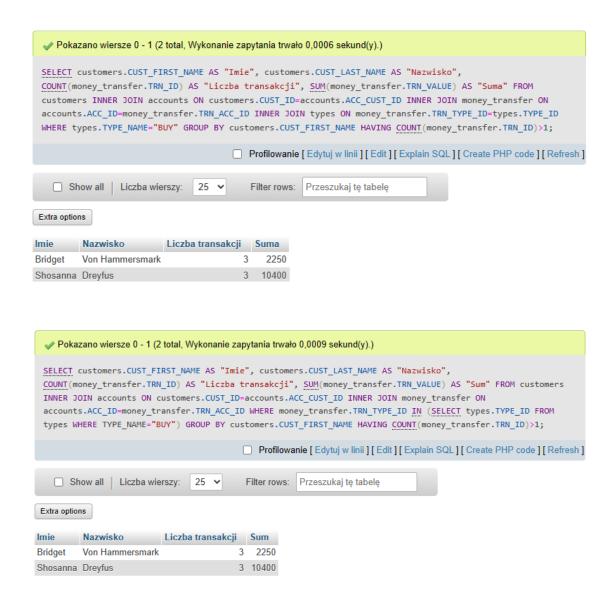
3. Napisać zapytanie tworzące prosty raport złożony z 2 kolumn dla wszystkich tabel w bazie. Raport powinien zawierać liczbę wierszy dla każdej z tabel oraz największe id.



4. Napisać zapytanie wyświetlające numery, statusy i typy kont oraz średnią wartość transakcji dla tych kont. Wykorzystać polecenia CASE WHEN oraz IF tak, aby statusy i typy kont nie były liczbami.



5. Napisać 2 zapytania wyświetlające imię, nazwisko klienta, liczbę wykonanych transakcji (min. 2 wykonane transakcje) oraz sumę ich wartości, tylko dla typu "BUY". Wymagania: 1 query – powiązanie tabeli "Types" instrukcją JOIN, 2 query - powiązanie tabeli "Types" zapytaniem zagnieżdżonym, W sprawozdaniu załączyć czasy wykonania obu zapytań.



Zapytanie wykorzystujące JOIN było szybsze, lecz różnica jest niewielka.

6. Utworzyć raport przy pomocy SQL, w którym będzie wyświetlana liczba wszystkich otwartych kont oraz ostatnia data transakcji dla otwartego konta, gdzie kod typu transakcji to "222". Wymagane instrukcje: WITH, CONCAT, UNION, filtrowanie pola JSON.

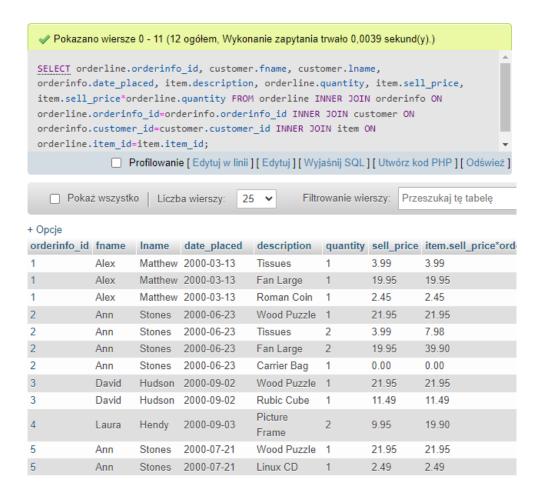
```
Uruchom zapytanie SQL/zapytania w tabeli finansowa_1.money_transfer: 🔞
  1 WITH otwarte_konta AS (
       SELECT
       FROM accounts
       WHERE ACC STATUS=1
 5),
  6 ostatnie_transakcje AS (
       SELECT
       money_transfer.TRN_ACC_ID AS id_konta,
      MAX(money_transfer.TRN_DATE) AS ostatnia_transakcja,
 9
      money_transfer.TRN_TYPE_ID AS typ
 10
 11
       FROM money_transfer
 12
       GROUP BY money_transfer.TRN_ACC_ID
 13 )
       CONCAT("Liczba otwartych kont: ",COUNT(ACC_ID)) AS ""
15
       FROM otwarte_konta
16
17
      SELECT CONCAT("Ostatnia transakcja konta nr: ", ostatnie_transakcje.id_konta, " - ",
   ostatnie_transakcje.ostatnia_transakcja)
      FROM ostatnie_transakcje
      INNER JOIN otwarte_konta ON ostatnie_transakcje.id_konta=otwarte_konta.ACC_ID
       INNER JOIN types ON ostatnie_transakcje.typ=types.TYPE_ID
 22
       WHERE JSON_EXTRACT(types.TYPE_DETAILS, "$.code") = '222'
 23
```

Liczba otwartych kont: 5 Ostatnia transakcja konta nr: 15 - 2023-04-16 Ostatnia transakcja konta nr: 11 - 2023-04-03

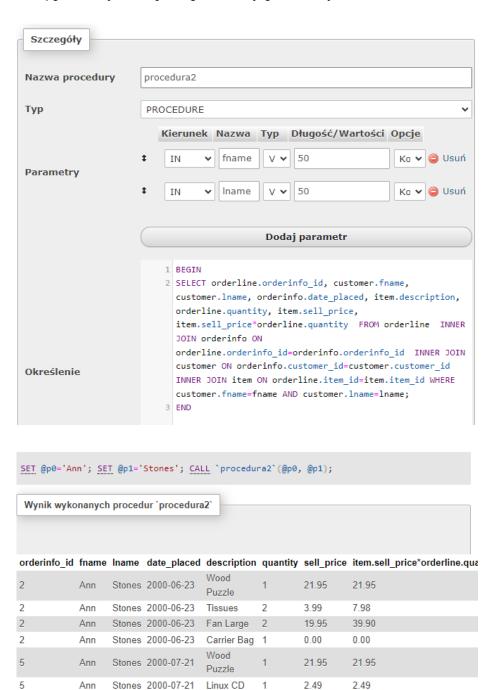
Baza danych

Sprzedaż Wysyłkowa

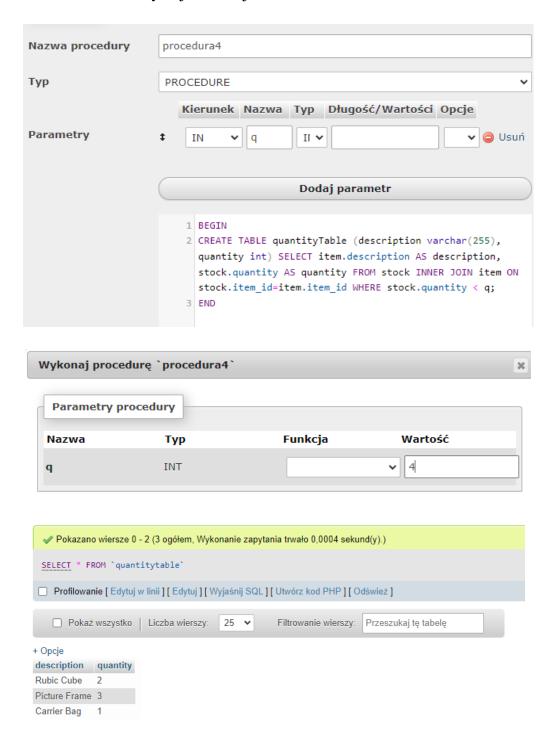
1. Utworzyć widok dla złożonych zamówień: numer zamówienia, imię, nazwisko, data zamówienia, zamówiony towar, liczba sztuk, cena sprzedaży, wartość zamówionego towaru (nie uwzględniać kosztów wysyłki)



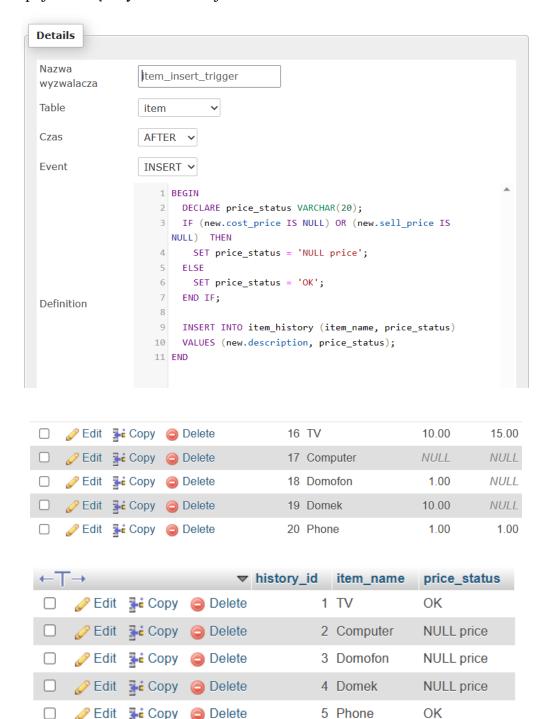
2. Utworzyć procedurę składowaną, która po uruchomieniu wyświetla następujące informacje: numer zamówienia, imię, nazwisko, data zamówienia, zamówiony towar, liczba sztuk, cena sprzedaży, wartość zamówionego towaru (nie uwzględniać kosztów wysyłki) ale tylko dla wybranego klienta, którego dane są przekazywane jako parametry procedury.



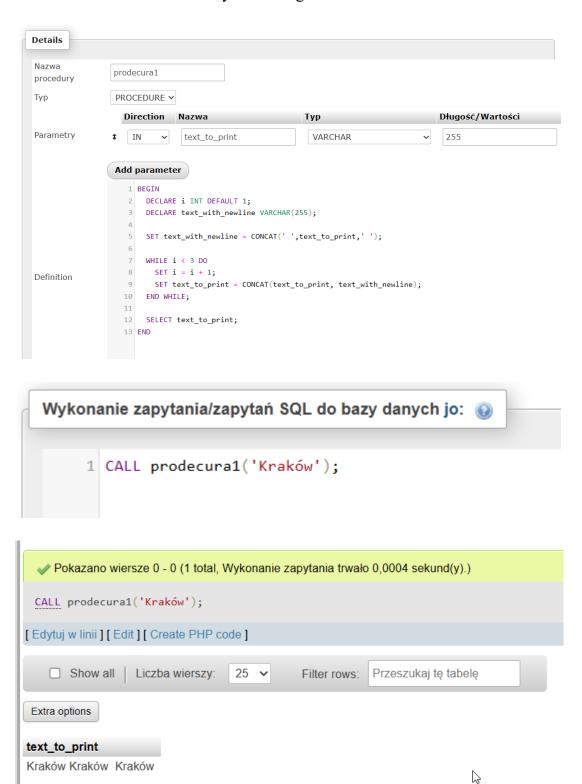
3. Utworzyć procedurę składowaną tworzącą tabelę z informacjami o towarach, których w tabeli stock jest mniej niż wartość zadanej liczby (będącej parametrem procedury). Działanie procedury zademonstrować dla towarów, których jest mniej niż 4 sztuki.



4. Utworzyć trigger (typu AFTER) dla INSERT dla tabeli item. Trigger powinien wypełniać rekordy w tabeli item_history. Standardowo, po wstawieniu nowego towaru przy użyciu INSERT powinna pojawiać się nazwa tego towaru oraz OK. Ale, gdy ceny zawierają NULL, zamiast OK, powinna pojawić się o tym informacja.



5. Utworzyć procedurę przechowywaną wypisującą przekazywany tekst trzy razy (w jednej linii): użyć konstrukcji LOOP, działanie procedury zademonstrować dla nazwy ulubionego miasta.



6. Załóżmy, że dane są dwie tabele dwukolumnowe. Pierwsze kolumny są identyczne i stanowią dla obu tabel klucz podstawowy. Drugie kolumny zawierają dowolne liczby całkowite. Przyjmijmy, że mamy po pięć rekordów. Należy napisać procedurę przechowywaną z kursorem tworzącym trzecia dwukolumnowa tabelę. W trzeciej tabeli pierwsza kolumna powinna być taka sama, jak w tabelach wyjściowych. Do drugiej kolumny wstawiamy tekst zależny od tego, która liczba większa w drugich kolumnach tabel wyjściowych: t1>t2, albo t1<=t2.

