Spring Data Access - Projekt

Spis treści

Opis

W ramach tego projektu przygotujemy schemat bazy danych oraz zasilimy go przykładowymi danymi w aplikacji, która zapisuje transakcje użytkowników w sklepie internetowym. Następnie pobawimy się implementacją przedstawionych zagadnień. Oczywiście jest to bardzo uproszczony sklep internetowy, taki na potrzeby edukacyjne ☺.

Baza danych, w której będziemy przetrzymywać te dane będzie się nazywała `zajavka_store` i będzie to baza danych analogiczna do bazy we wcześniej poruszanym projekcie **zajavka store**.

W naszym sklepie chcemy mieć następujące tabele:

- **CUSTOMER** - tabela przechowująca informacje o klientach sklepu, chcemy by przetrzymywała następujące informacje:
  - **ID** - unikalny identyfikator użytkownika w naszym sklepie, klucz główny,
  - **USER_NAME** - unikalna nazwa użytkownika w naszym sklepie, unikalne, not null,
  - **EMAIL** - unikalny email użytkownika w naszym sklepie, unikalne, not null,
  - **NAME** - imię użytkownika, not null,
  - **SURNAME** - nazwisko użytkownika, not null,
  - **DATE_OF_BIRTH** - data urodzenia, nullable,
  - **TELEPHONE_NUMBER** - numer telefonu, nullable,
- **PRODUCER** - tabela z informacjami o producentach produktów w naszym sklepie, chcemy by przetrzymywała następujące informacje:
  - **ID** - unikalny identyfikator producenta, klucz główny,
  - **PRODUCER_NAME** - unikalna nazwa producenta, unikalne, not null,
  - **ADDRESS** - adres producenta wpisany w jednej linijce, nullable,
- **PRODUCT** - tabela z produktami oferowanymi w naszym sklepie, chcemy by przetrzymywała następujące informacje:
  - **ID** - unikalny identyfikator produktu, klucz główny,
  - **PRODUCT_CODE** - unikalny kod produktu, coś w stylu kod kreskowego, unikalne, not null,
  - **PRODUCT_NAME** - nazwa produktu, not null,
  - **PRODUCT_PRICE** - cena produktu, not null,

    - **ADULTS_ONLY** - flaga mówiąca, czy produkt może być sprzedawany tylko osobom, które ukończyły 18 lat, not null,
    - **DESCRIPTION** - opis produktu, not null,
    - **PRODUCER_ID** - informacja o producencie produktu, klucz obcy, not null,
- **PURCHASE** - tabela z informacjami o tym, kto i jaki produkt zakupił oraz kiedy i w jakiej ilości chcemy by przetrzymywała następujące informacje:
    - **ID** - unikalny identyfikator zakupu, klucz główny,
    - **CUSTOMER_ID** - informacja o tym który klient dokonał zakupu, klucz obcy, not null,
    - **PRODUCT_ID** - informacja o zakupionym produkcie, klucz obcy, not null,
    - **QUANTITY** - ilość zakupionych produktów, not null,
    - **DATE_TIME** - data i czas dokonania zakupu, not null,
- **OPINION** - tabela z opiniami użytkowników dotyczącymi produktów w sklepie chcemy by przetrzymywała następujące informacje:
    - **ID** - unikalny identyfikator opinii, klucz główny,
    - **CUSTOMER_ID** - informacja o tym który klient dodał opinię, klucz obcy, not null,
    - **PRODUCT_ID** - informacja o tym jakiego produktu dotyczy opinia, klucz obcy, not null,
    - **STARS** - ocena produktu, może przyjąć tylko wartości ze zbioru (1, 2, 3, 4, 5), not null,
    - **COMMENT** - komentarz, not null,
    - **DATE_TIME** - data i czas dodania opinii, not null.

Zagadnienia

Sporą część opisanych poniżej przypadków możesz zrealizować jako przypadek testowy, czyli napisać do każdego zagadnienia oddzielny JUnit test.

1. Stwórz wspomnianą bazę danych.
2. Stwórz w bazie danych opisane tabele.
3. Napisz program w oparciu o Spring Data Access, w którym przygotujesz możliwość zasilenia wspomnianych tabel losowymi danymi. Przykładowo możesz stworzyć serwis, który będzie tworzył losowo wygenerowane dane, a następnie program będzie mógł zapisać te dane w bazie danych.
4. Przećwiczmy zarządzanie transakcjami. Stwórz program, który usuwa klienta ze sklepu, który ma w tym sklepie już jakąś historię, np. dokonał zakupów, czy wystawił jakąś opinię do produktu. Stwórz sytuację, która będzie wymagała użycia transakcji, np. możemy usunąć klienta tylko gdy jego wiek nie przekracza 40 lat (zastanów się jak można tutaj napisać problem, żeby taka transakcja była konieczna do poprawnego działania). Upewnij się, że jeżeli usuwanie danych z jednej tabeli się nie powiedzie to transakcja nie jest commitowana i dane są nadal spójne.
5. Dodaj sprawdzenie, czy klient wystawia ocenę do produktu, który faktycznie został przez niego zakupiony.
6. Dodaj możliwość załadowania danych do bazy na podstawie SQL wczytanych z pliku. Plik taki umieść w katalogu `src/main/resources`. Przykładowy plik z insertami został umieszczony w poście

obok treści zadania. Do stworzenia pliku z insertami została wykorzystana strona https://www.mockaroo.com/.

Chciałbym żeby każde z poniższych poleceń rozpoczynało się od pełnego odtworzenia stanu bazy danych na podstawie dostarczonego pliku z insertami.

7. Nie podobają nam się opinie o ocenie niższej niż 4, więc usuń je wszystkie 😈.
8. Nie podobają nam się klienci, którzy wystawiają oceny niższe niż 4 więc usuń ich wszystkich. Pamiętaj o wcześniejszym ograniczeniu w usuwaniu klientów.
9. Istniejący klient dokonuje zakupu w sklepie. Dopisz logikę w kodzie, która umożliwi zrobienie zakupu w sklepie i napisz do tego przypadek testowy.
10. Chcemy zatrzeć ślad o istnieniu wybranego produktu. Napisz logikę, która na to pozwoli oraz przypadek testowy.