Projektowanie aplikacji z bazami danych

lista zadań nr 2

Wszystkie zadania są oparte o bazę AdventureWorksLT.

1. Utworzyć zapytanie, które na podstawie SalesOrderHeader.ShipToAddressID zwróci listę miast, do których towary zostały już dostarczone. Lista ma być posortowana i bez powtarzających się wartości.
   [1p]

2. Utworzyć zapytanie, które w wyniku zwróci dwie kolumny: nazwę modelu produktu (ProductModel.Name) oraz liczbę produktów tego modelu, przy czym w wyniku chcemy widzieć tylko te, dla których ta liczba jest większa niż 1. Zastanowić się, jakie konsekwencje rodzi fakt wyboru nazwy jako wartości grupującej.
   [1p]

3. Utworzyć zapytanie, które w wyniku zwróci trzy kolumny: nazwę miasta (z tabeli Address), liczbą klientów z tego miasta, liczbą SalesPerson obsługujących klientów z tego miasta.
   [1p]

4. Kategorie produktów są w strukturze drzewa. Możemy oczekiwać, że wszystkie produkty będą przypisane tylko do kategorii będących w liściach tego drzewa. Utworzyć zapytanie, które zwróci dwie kolumny: nazwę kategorii i nazwę produktu dla produktów będących przypisanych do kategorii nie będących w liściach.
   [1p]

5. Utworzyć zapytanie, który w wyniku zwróci trzy kolumny: nazwisko i imię klienta (Customer) oraz kwotę, którą ten klient zaoszczędził dzięki udzielonym rabatom (SalesOrderDetail.UnitPriceDiscount).
   [1p]

6. Utworzyć tabelę OrdersToProcess(SalesOrderID INT, Delayed BIT), w której będą przechowywane zamówienia jeszcze nie dostarczone, a flaga Delayed określa czy DueDate nie został przekroczony. Przygotować zapytanie, które zaktualizuje tę tabelę w oparciu o tabelę SalesOrderHeader i należy skorzystać z konstrukcji MERGE. Aby przetestować działanie tej metody, warto dogenerować trochę danych (w bazie testowej jest ich niewiele).
   [1p]

7. Utwórz tabelę Test z kolumną IDENTITY, gdzie identyfikatory mają się zaczynać od 1000 i przesuwać o 10. Zademonstruj różnicę pomiędzy @@IDENTITY i IDENT_CURRENT.
   [1p]

8. Zapoznać się z ograniczeniem (constraint) SalesOrderHeader.CK_SalesOrderHeader_ShipDate, zaprezentować instrukcję jego utworzenia. Spróbować dodać wiersz (lub zmodyfikować istniejący) naruszając to ograniczenie. Jaki będzie efekt? Następnie wyłączyć ogranicznie i spróbować ponownie. Na koniec włączyć ograniczenie i wylistować bieżące naruszenia.
   [1p]

9. Do tabeli Customer dodaj kolumnę CreditCardNumber, a w tabeli SalesOrderHeader dla 3 wybranych pozycji ustaw dowolną wartość dla kolumny CreditCardApprovalCode. Następnie, klientom (Customer), którzy mają zamówienia (SalesOrderHeader) z ustawionym CreditCardApprovalCode, zmień pole CreditCardNumber na wartość 'X'.
   [1p]

10. Utworzyć tabele *M1(K INT, V VARCHAR(20))* oraz *S1(K INT, MFK INT, V VARCHAR(20))*, gdzie *K* jest kluczem głównym, a *MFK* jest kluczem obcym do tabeli *M*. Następnie utworzyć *M2* oraz *S2* z tą różnicą, że klucz główny *M2* składa się z dwóch kolumn: *K1* oraz *K2*, a *S2* ma odpowiednio dopasowany klucz obcy. Dodać trochę danych i sprawdzić, że relacja klucza obcego istotnie działa. Ostatecznie, dodać klauzule *ON UPDATE* oraz *ON DELETE* i zademonstrować działanie relacji klucza obcego dla wartości *NO ACTION*, *SET NULL* oraz *CASCADE*.
    [1p]

Uwaga: Jeśli ktoś nie będzie miał pomysłu na zrobienie zadania jednym zapytaniem, może je zrobić kilkoma zapytaniami, gdzie częściowe wyniki będą przechowywane w dodatkowo utworzonych tabelach.

Paweł Rajba