System zakupów internetowych

Autor: Robert Czwartosz 226296

Kurs: Rozproszone i obiektowe bazy danych

Data: 17.01.2020r.

Cel i zakres zadania

Celem zadania było stworzenie systemu zakupów internetowych przy użyciu rozproszonych baz danych. Bazy danych są zaimplementowane przy użyciu systemu zarządzania **Oracle 11g**. Aplikacja dostępowa jest zaimplementowana w języku **Python 3**.

Funkcjonalny opis działania systemu

System umożliwia dostęp do konta administratora oraz kont klientów(użytkowników). Administrator ma możliwość przeglądania i filtrowania użytkowników, pracowników, produktów oraz złożonych zamówień. Administrator również może edytować produkty, pracowników i użytkowników; dodawać produkty i pracowników oraz usuwać pracowników, użytkowników i zamówienia. Każde zamówienie ma status (przyjęte, realizowane lub wykonane), który jest zmieniany przez administratora. Status informuje użytkownika o aktualnym stanie zamówienia.

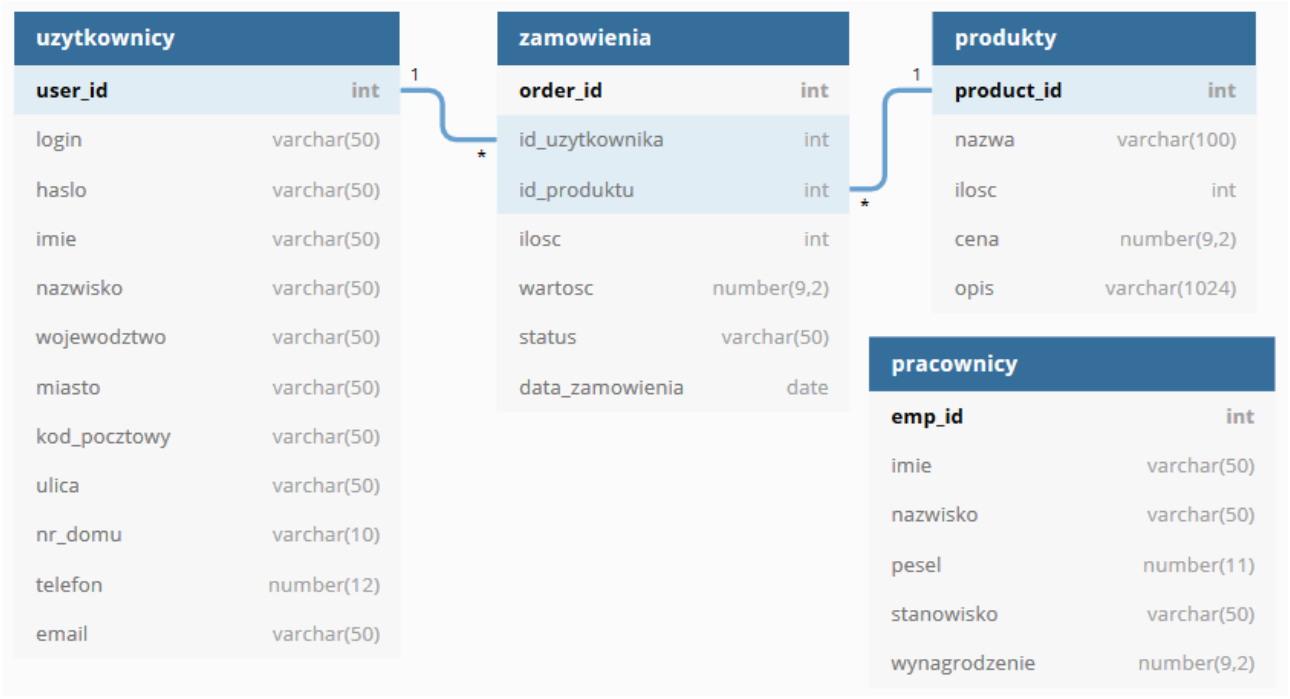
Każdy kto chce złożyć zamówienie w sklepie internetowym powinien założyć konto oraz zalogować się na nie. Po zalogowaniu użytkownik ma możliwość przeglądania produktów oraz składania i anulowania zamówień.

Założenia projektowe

Działanie systemu jest oparte na komunikacji klient-serwer. Wykorzystano architekturę trójwarstwową (warstwa prezentacji, warstwa logiki biznesowej, warstwa danych). Zastosowano następujące mechanizmy rozproszenia danych: fragmentacja pionowa, fragmentacja mieszana, replikacja master-slave, replikacja master-master (mechanizm *Oracle streams*). Przetwarzanie danych polega na wykonywaniu odpowiednich **procedur** oraz mechanizmu ***Oracle streams*** [1][2].

Opis implementacji bazy danych

Z punktu widzenia użytkownika rozproszonej bazy danych, baza składa się z czterech tabel: *uzytkownicy*, *produkty*, *zamowienia* i *pracownicy*.



Rysunek 1: Diagram UML rozproszonej bazy danych

Tabela *uzytkownicy* jest partycjonowana poziomo (na podstawie pola *wojewodztwo*) i pionowo, zatem jest to **fragmentacja mieszana**.

```
CREATE VIEW robert.uzytkownicy
AS
SELECT UZYTKOWNICY_DOL_PASS.HASLO, UZYTKOWNICY_DOL_DATA.*
FROM UZYTKOWNICY_DOL_PASS FULL JOIN UZYTKOWNICY_DOL_DATA
ON UZYTKOWNICY_DOL_DATA.login = UZYTKOWNICY_DOL_PASS.login
UNION
SELECT UZYTKOWNICY_NDOL_PASS.HASLO, UZYTKOWNICY_NDOL_DATA.*
FROM UZYTKOWNICY_NDOL_PASS FULL JOIN UZYTKOWNICY_NDOL_DATA
ON UZYTKOWNICY_NDOL_DATA.login = UZYTKOWNICY_NDOL_PASS.login;
```

Tabela *zamowienia* jest **partycjonowana poziomo**(na podstawie województwa w którym jest użytkownik zamawiający).

```
CREATE VIEW robert.zamowienia
AS
SELECT * from robert.zamowienia_dol
UNION
SELECT * from robert.zamowienia_ndol;
```

Tabela *produkty* jest **replikowana** przy użyciu **perspektywy zmaterializowanej** odświeżanej w sposób przyrostowy co 10s.

```
CREATE MATERIALIZED VIEW robert.produkty
  BUILD IMMEDIATE
  REFRESH FAST
  NEXT sysdate+(1/(24*60*6))
  WITH PRIMARY KEY
AS
SELECT * FROM robert.produkty@orcl;
```

Tabela *pracownicy* jest **replikowana** przy użyciu mechanizmu ***Oracle streams*** [1][2]. Replikacja przy użyciu tego mechanizmu przestaje działać po ponownym uruchomieniu systemu. Aby przywrócić działanie replikacji należy usunąć w obu bazach użytkownika *stradmin* przy użyciu polecenia: *DROP USER stradmin CASCADE;*. Następnie należy skopiować zawartość tabeli poleceniem: *create table pracownicy_kopia as select * from pracownicy;*. Po skopiowaniu należy usunąć i ponownie stworzyć tabele w dwóch bazach poleceniami *drop* i *create table*. Po wykonaniu tych czynności trzeba na nowo utworzyć w obu bazach użytkownika *stradmin*, kolejki *apply* i *capture* oraz procesy replikacji. Ostatnią czynnością jest dodanie rekordów ze skopiowanej tabeli przy użyciu polecenia *insert … select*.

Uruchamianie i testowanie bazy danych

Aby uruchomić rozproszoną bazę danych należy uruchomić wirtualną maszynę, na której jest baza ORCL1. Następnie należy uruchomić usługi związane z systemem Oracle na dwóch systemach operacyjnych, w ten sposób bazy ORCL i ORCL1 są aktywne. Po wykonaniu tych czynności można połączyć się z jedną z dwóch baz i przejść do testowania. Test powinien zakończyć się powodzeniem bez względu na to do której bazy się połączono.

Aby przetestować rozproszoną bazę danych wykonywano następujące czynności:

- wykonanie procedury *insert_user* z argumentem *wojewodztwo='Dolnośląskie'*, a następnie sprawdzenie czy użytkownik jest dodawany do tabel *uzytkownicyDOL_data* oraz *uzytkonicyDOL_pass*
- wykonanie procedury *insert_user* z argumentem *wojewodztwo='Wielkopolskie'*, a następnie sprawdzenie czy użytkownik jest dodawany do tabel *uzytkownicyNDOL_data* oraz *uzytkonicyNDOL_pass*
- wykonanie polecenia *select * from uzytkownicy* i sprawdzenie, czy wyświetlają się użytkownicy ze wszystkich województw oraz czy wyświetlane są hasła do ich kont.
- Wykonanie procedury *delete_user* i sprawdzenie czy użytkownik został usunięty z widoku *uzytkownicy*

- wykonanie procedury *insert_product* i sprawdzenie czy produkt jest dodawany do tabeli *produkty* oraz do jej replikacji znajdującej się w bazie ORCL1
- wykonanie procedury *dostawa_produktu* i sprawdzenie czy zwiększyła się ilość danego produktu w tabeli *produkty* oraz w jej replikacji znajdującej się w bazie ORCL1
- wykonanie procedury *dodaj_zamowienie* dla użytkownika z województwa dolnośląskiego i sprawdzenie czy zamówienie zostało dodane do tabeli *zamowienia_DOL*
- wykonanie procedury *dodaj_zamowienie* dla użytkownika z województwa wielkopolskiego i sprawdzenie czy zamówienie zostało dodane do tabeli *zamowienia_NDOL*
- wykonanie polecenia *select * from zamowienia* i sprawdzenie czy wyświetlają się wszystkie dodane zamówienia
- wykonanie procedury *zmien_status* i sprawdzenie czy zmienił się status danego zamówienia w widoku *zamowienia*
- wykonanie procedury *anuluj_zamowienie* i sprawdzenie czy zamówienie zostało usunięte z widoku *zamowienia*
- wykonanie procedury *dodaj_pracownika* i sprawdzenie czy pracownik został dodany do tabeli *pracownicy* w bazach ORCL i ORCL1
- wykonanie procedury *usun_pracownika* i sprawdzenie czy dany pracownik został usunięty z tabeli *pracownicy* w bazach ORCL i ORCL1

Opis implementacji aplikacji dostępowej

Aplikacja dostępowa jest aplikacją webową napisaną w języku Python 3. Framework **django** jest użyty do implementacji serwera aplikacyjnego, który wysyła zapytania do bazy danych [3]. Framework umożliwia tworzenie szablonów na podstawie których renderowana jest strona w języku **HTML**. W szablonach użyto biblioteki **Bootstrap** [4]. Aplikacja dostępowa realizuje funkcjonalności opisane wcześniej.

Podsumowanie

System zakupów internetowych został zrealizowany wg wcześniejszych założeń. Tworzenie systemu rozproszonych baz danych jest znacznie trudniejsze niż tworzenie zcentralizowanego systemu baz danych. Jednak zaletami tego systemów są: szybsze wyszukiwanie danych (zapewnione przez fragmentację poziomą), mniejsze współzawodnictwo procesów o dostęp do danych oraz mniejsza awaryjność systemu(zapewniona przez replikację) [1]. Zatem dla dużej liczby użytkowników systemu oraz dużej ilości danych zastosowanie systemu rozproszonego będzie konieczne, pomimo że wykrywanie i usuwanie skutków awarii oraz zarządzanie bezpieczeństwem systemu są znacznie utrudnione [1].

Bibliografia

[1] http://robert.wojcik.staff.iiar.pwr.wroc.pl/dydaktyka/dzienne/rsbd/RB_6.pdf, dostęp: 16.01.2020

[2] https://www.youtube.com/watch?v=GWF7uErEm4k, dostęp 16.01.2020

[3] https://docs.djangoproject.com/en/3.0/intro/ dostęp 16.01.2020

[4] https://getbootstrap.com/, dostęp 16.01.2020