Relacyjne Bazy Danych

Andrzej M. Borzyszkowski
PJATK/ Gdańsk

materiały dostępne elektronicznie
http://szuflandia.pjwstk.edu.pl/~amb

Język SQL, cz. 2, operowanie na danych (*data manipulation language*) (uzupełnienia)

Perspektywy

- Bardziej skomplikowane zapytanie może zostać zapamiętane

```
CREATE VIEW towar_zysk AS
  SELECT *, cena – koszt AS zysk FROM towar
```

  - zapamiętuje pytanie
  - późniejsze użycie odnosi się do treści tabeli z momentu tego użycia

```
SELECT * FROM towar_zysk
```

  - jest zawsze równoważne zapytaniu

```
SELECT *, cena – koszt AS zysk FROM towar
```


Perspektywy a tabele tymczasowe

- Perspektywa jest czym innym niż tabela tymczasowa

```
CREATE TEMP TABLE towar_zysk (
  nr      int PRIMARY KEY,
  opis    varchar(64) ,  koszt   numeric(7,2) ,
  cena    numeric(7,2) ,  zysk    numeric(7,2) )
INSERT INTO towar_zysk
  SELECT *, cena – koszt AS zysk FROM towar
```

  - wstawia do utworzonej wcześniej tabeli wynik obliczenia operacji SELECT
  - po zmianie zawartości tabeli towarów pytania

```
SELECT *, cena – koszt AS zysk FROM towar
SELECT * FROM towar_zysk
```

  - dadzą *różne* odpowiedzi, nową i starą wartość zysku

Perspektywy c.d.

- Tabela tymczasowa może być używana tak jak każda tabela, w szczególności można do niej wstawiać i z niej usuwać krotki
- Operacje UPDATE i DELETE dla perspektyw nie są oczywiste

```
DELETE from towar_zysk
WHERE zysk/koszt<0.05
```

  - można sobie wyobrazić realizację powyższego polecenia jako

```
DELETE from towar
WHERE (cena-koszt)/koszt<0.05
```

  - ale jak miałaby działać poniższa operacja na tabeli towar?

```
UPDATE towar_zysk
SET zysk=zysk*1.1
```


Perspektywy 3.

- PostgreSQL do wersji 9.2 nie przewidywał operacji UPDATE i DELETE dla perspektyw

```
amb=> create view test as select * from towar;
CREATE VIEW
amb=> delete from test where nr=6;
ERROR:  cannot delete from view "test"
PODPOWIEDŹ:  You need an unconditional ON DELETE DO INSTEAD
rule or an INSTEAD OF DELETE trigger.
```

  - od wersji 9.3 dla szczególnie prostych perspektyw, definiowanych w oparciu o pojedynczą tabelę, bez grupowania, można używać operacje INSERT, DELETE i UPDATE
  - od wersji 9.4 perspektywa można dopuszczać pewne kolumny bez możliwości aktualizacji podczas gdy inne z możliwością
  - inne systemy zarządzania bazami danych mają/mogą mieć pewne możliwości operowania na perspektywach

Instrukcja SELECT – zagnieżdżenie

- Podaj dane klientów którzy złożyli zamówienia po 1 marca 2021

```
SELECT nazwisko FROM klient
WHERE nr IN ( SELECT klient_nr
              FROM zamowienie
              WHERE data_zlozenia > '2021-3-1'
            )
```

  - zagnieżdżona tabela użyta w warunku służy jako zbiór
  - możemy najpierw wykonać wewnętrzne zapytanie, a potem zewnętrzne
  - *brak* korelacji, w zapytaniu podrzędnym nie ma odwołania do wiersza z tabeli zewnętrznej

Zagnieżdżenia: korelacja

- Podaj dane klientów, których nazwiska się powtarzają:

```
SELECT imie, nazwisko, miasto
FROM klient
WHERE nazwisko IN (
   SELECT nazwisko
   FROM klient
   GROUP BY nazwisko HAVING count (nazwisko) > 1
   )
```

  - *brak* korelacji, w zapytaniu podrzędnym nie ma odwołania do wiersza z tabeli zewnętrznej
  - mimo, że ta sama tabela przeglądana jest dwukrotnie, brak korelacji powoduje brak potrzeby zmiany nazwy
  - wiadomo w każdym miejscu o czyje nazwisko chodzi

Zagnieżdżenia: korelacja c.d.

- Podaj dane klientów, którzy złożyli zamówienie po 1 marca 2021

```
SELECT nazwisko
FROM klient
WHERE EXISTS (
  SELECT *
  FROM zamowienie
  WHERE klient.nr = klient_nr AND data_zlozenia > '2021-3-1'
)
```

  - wewnętrzny SELECT odwołuje się do tabeli zewnętrznej
  - *nie możemy* wykonać wewnętrznego zapytania w oderwaniu od reszty
  - *występuje* korelacja
  - moglibyśmy użyć aliasu dla tabeli zewnętrznej, ale konieczności nie ma

Zagnieżdżenia: korelacja 3.

- Podaj dane klientów, których nazwiska się powtarzają:

```
SELECT imie, nazwisko, miasto
FROM klient K
WHERE EXISTS (
    SELECT *
    FROM klient
    WHERE  nazwisko=K.nazwisko AND nr != K.nr
    )
```

  - *występuje* korelacja, wewnętrzne pytanie zawiera odwołanie do wiersza z tabeli zewnętrznej
  - dodatkowo, tutaj tabela przeglądana w podrzędnym zapytaniu jest ta sama co w zewnętrznym, występuje konieczność nazwania zewnętrznej tabeli