Studia podyplomowe ”Inżynieria oprogramowania” współfinansowane przez Unię Europejską w ramach Europejskiego Funduszu Społecznego

Projekt ”Studia podyplomowe z zakresu wytwarzania oprogramowania oraz zarządzania projektami w firmach informatycznych” realizowany w ramach Programu Operacyjnego Kapitał Ludzki

Konstruowanie Baz Danych

SQL — UNION, INTERSECT, EXCEPT

Antoni Ligęza
ligeza@agh.edu.pl
http://home.agh.edu.pl/~ligeza
http://home.agh.edu.pl/~ligeza/wiki

Bazy danych i systemy zarządzania

Wykład VIII

Elementy języka SQL

Część II

Zagnieżdżanie zapytań

Zapytania w SQL mogą operować na wynikach innych zapytań; możliwe jest więc tzw. zagnieżdżanie zapytań. Typowy schemat zagnieżdżania:

```
SELECT Kolumna/Kolumny
   FROM Tabela
   WHERE  Atrybut IN|NOT IN|[[=,<,>,...]ANY|ALL]
    SELECT Kolumna/Kolumny
                          FROM Tabela
                          WHERE Warunek;
```

```
SELECT DataZlozeniaZamowienia, NumerKlienta
   FROM Zamowienia
   WHERE NumerZamowienia IN (
      SELECT NumerZamowienia
         FROM ZamowioneKsiazki
         WHERE ISBN = '83-204-1806-2'
                           );
```

```
SELECT Nazwisko, Imie
   FROM Klienci
   WHERE NumerKlienta IN (
      SELECT NumerKlienta
         FROM Zamowienia
         WHERE NumerZamowienia = ANY (
            SELECT Numer Zamowienia
               FROM ZamowioneKsiazki
               WHERE Rok = '1999'
                        )
                                    );
```

Operatory `IN` oraz `ANY` pozwalają na przeszukanie pewnego zbioru; różnica pomiędzy nimi polega na możliwości zastosowania różnych operatorów relacyjnych (zamiast "=") dla `ANY`.

Złączenie

W specyfikacji tabeli, po słowie FROM można zdefiniować dowolne złączenie tabel (naturalne, θ-złączenie, iloczyn kartezjański; wewnętrzene, zewnętrzne i typu *union*). Składnia odpowiednich wyrażeń jest następująca.

- złączenie krzyżowe: TablicaA `CROSS JOIN` TablicaB,
- złączenie naturalne: TablicaA `[NATURAL]` [typ złączenia] `JOIN` TablicaB,
- złączenie union: TablicaA `UNION JOIN` TablicaB,
- złączenie przez predykat : TablicaA [typ złączenia] `JOIN` TablicaB `ON` predykat,
- złączenie po zadanej kolumnie: TablicaA [typ złączenia] `JOIN` TablicaB `USING` (nazwa kolumny,...),
- typy złączenia: `INNER`| `{LEFT|RIGHT|FULL} [OUTER]`.

Przykłady:

```
SELECT Imie, Nazwisko, NumerZamowienia, DataZamowienia
   FROM Klienci, Zamowienia
   WHERE Klienci.NumerKlienta = Zamowienia.NumerKlienta
      AND Nazwisko IN ('Kowalski', 'Malinowski', 'Nowak');

SELECT Imie, Nazwisko, NumerZamowienia, DataZamowienia
   FROM Klienci JOIN Zamowienia
   ON Klienci.NumerKlienta = Zamowienia.NumerKlienta;

SELECT Imie, Nazwisko, NumerZamowienia, DataZamowienia
   FROM Klienci T1 LEFT OUTER JOIN Zamowienia T2
   ON (T1.NumerKlienta = T2.NumerKJlienta);
```

Suma relacji zgodnych – operator `UNION` wewnątrz klauzuli `SELECT`

```
SELECT kolumnaA1, kolumnaA2,..., kolumnaAk
   FROM tabelaA
   WHERE predykatA
UNION
SELECT kolumnaB1, kolumnaB2,..., kolumnaBk
   FROM tabelaB
   WHERE predykatB;
```

Wskazane kolumny muszą mieć identyczne typy i rozmiary (niekoniecznie nazwy) oraz musi być zachowana zgodna kolejność tych kolumn. Zastosowanie operatora `UNION` automatycznie usuwa duplikaty.

```
SELECT Identyfikator, Nazwisko, Imie, Wiek
   FROM Pracownicy
   WHERE Wiek < 28
UNION
SELECT Identyfikator, Nazwisko, Imie, Wiek
   FROM Studenci
   WHERE Wiek < 28;
```

Opcja `CORRESPONDING BY` jest określona dla SQL-92 (gdy wybrane kolumny są zgodne, co do nazwy; struktura tabel nie musi być identyczna).

```
SELECT *
   FROM Pracownicy
   WHERE Wiek < 28
UNION CORRESPONDING BY (Identyfikator, Nazwisko, Imie, Wiek)
SELECT *
   FROM Studenci
   WHERE Wiek < 28;
```

Kolumny nie wykazane w opcji `CORRESPONDING BY` są pomijane.

Suma relacji zgodnych – operator `OR` w opcji `WHERE` klauzuli `SELECT`

```
SELECT kolumna1, kolumna2,..., kolumnak
   FROM Tabela
   WHERE PredykatA
UNION
SELECT kolumna1, kolumna2,..., kolumnak
   FROM Tabela
   WHERE PredykatB;
```

Dla realizacji sumy elementów tej samej tabeli można wykorzystać spójnik `OR`:

```
SELECT kolumna1, kolumna2,..., kolumnak
   FROM Tabela
   WHERE PredykatA OR PredykatB;
```

```
SELECT Identyfikator, Nazwisko, Imie, Wiek
   FROM Pracownicy
   WHERE Dzial = 'Kadry'
UNION
SELECT Identyfikator, Nazwisko, Imie, Wiek
   FROM Pracownicy
   WHERE Dzial = 'Place';
```

może być zastąpione przez:

```
SELECT Identyfikator, Nazwisko, Imie, Wiek
   FROM Pracownicy
   WHERE Dzial = 'Kadry' OR Dzial = "Place';
```

Zapytanie z `UNION` dwukrotnie przeszukuje tabelę; `OR` nie usuwa duplikatów.

Suma relacji – polecenie `INSERT`

Polecenie `INSERT INTO` może być wykorzystane do dopisania pojedynczego wiersza do tabeli oraz skopiowania jednego lub wielu wierszy z innej tabeli; ma ono ogólną postać:

```
INSERT INTO Tabela
   [(kolumna1, kolumna2,...,kolumnak)]
   VALUES (lista wartosci);
```

lub

```
INSERT INTO TabelaA
   [(kolumnaA1, kolumnaA2,...,kolumnaAk)]
   SELECT kolumnaB1, kolumnaB2,...,kolumnaBk
      FROM TableaB
      WHERE WarunekWyboruWierszy;;
```

Przykładowe zastosowania:

```
INSERT INTO Ksiazki
   VALUES ('83-87102-55-5','Harrington','SQL dla kazdego',
           'EDU-Mikom', 1998, 'Warszawa');

INSERT INTO Ksiazki
   (ISBN,Autor,Tytul)
   VALUES ('83-87102-55-5','Harrington','SQL dla kazdego');

INSERT INTO Ksiazki
   SELECT ISBN, Autor, Tytul, Wydawnictwo, Rok, Miejsce
      FROM ZamowioneKsiazki
      WHERE Status = 'Dostarczone';
```

Uwaga: rekordy generowane poleceniem `SELECT` muszą być zgodne co do struktury (liczba, typ) z tabelą docelową.

Różnica (odejmowanie) tabel/relacji

```
SELECT kolumna1, kolumna2,..., kolumnak
   FROM TabelaA
   WHERE (kol1, kol2,...,kolj) NOT IN (SELECT kol1,...,kolj
                                       FROM TabelaB
                                       WHERE Warunek);
```

Przykłady:

```
SELECT Autor, Tytul
   FROM Ksiazki
   WHERE ISBN NOT IN (SELECT ISBN
                      FROM ZamowKsiazki);

SELECT NazwaWydawcy
   FROM Wydawcy
   WHERE NazwaWydawcy NOT IN (SELECT NazwaWydawcy
                              FROM Ksiazki
                              WHERE ISBN IN (SELECT ISBN
                                             FROM ZamowKsiazki)
      );

SELECT NazwaKlienta, NrKlienta
   FROM KLienci
   WHERE NrKlienta  NOT IN (SELECT NrKlienta
                            FROM Zamowienia
                            WHERE Data >= '2000-01-01');


SELECT IdFaktury, TypElementu, Odbiorca
 FROM RejestrSprzedazy
 WHERE (IdFaktury, TypElementu) NOT IN (SELECT IdFaktury,
                              TypElementu FROM Zaplacone);
```

Różnica z wykorzystaniem `EXCEPT`

```
SELECT kolumna1, kolumna2,...,kolumnak
   FROM TabelaA
   EXCEPT
   SELECT kolumna1, kolumna2,...,kolumnak
      FROM TabelaB;
```

```
SELECT *
   FROM TabelaA
   EXCEPT CORRESPONDING BY (kol1, kol2,...,kolk)
   SELECT *
      FROM TabelaB;
```

Przykład:

```
SELECT Autor, Tytul, RokWydania
   FROM Ksiazki
   EXCEPT
   SELECT Autor, Tytul, RokWydania
      FROM KsiazkiDoKasacji;
```

Różnicę można też zrealizować za pomocą złączenia `LEFT OUTER JOIN`.

```
SELECT kolumna1, kolumna2,...,kolumnak
   FROM (TabelaA LEFT OUTER JOIN TabelaB)
   ON TabelaA.Atrybut = TabelaB.Atrybut
   WHERE TabelaB.Atrybut IS NULL;
```

Przykład:

```
SELECT DISTINCT Ksiazki.*
   FROM (Ksiazki LEFT OUTER JOIN KsiazkiDoKasacji)
   ON TabelaA.Atrybut = TabelaB.Atrybut
   WHERE TabelaB.Atrybut IS NULL;
```

Iloczyn zwykły tabel/relacji

```
SELECT kolumna1, kolumna2,...,kolumnak
   FROM TabelaA
   INTERSECT
   SELECT kolumna1, kolumna2,...,kolumnak
   FROM TabelaB;
```

Przykłady:

```
SELECT Autor, Tytul, RokWydania
   FROM ZamowieniaA
   INTERSECT
   SELECT Autor, Tytul, RokWydania
   FROM ZamowieniaB;
```

```
SELECT *
   FROM ZamowieniaA
   WHERE RokWydania > '1995'
   INTERSECT CORRESPONDING  BY (Autor, Tytul, RokWydania)
   SELECT *
      FROM ZamowieniaB
      WHERE Cena < 1000;
```

Inne możliwości realizacji iloczynu: $R \cap S = R \setminus (R \setminus S)$ oraz `JOIN`.

```
SELECT Autor, Tytul, RokWydania
   FROM ZamowieniaA JOIN ZamowieniaB ON (Autor, Tytul, RokWydani
   WHERE ZamowieniaA.RokWydania > '1995' AND
         ZamowieniaB.Cena < 100;
```

Usuwanie i modyfikacja rekordów

Kasowanie rekordów

```
DELETE FROM Tabela
   WHERE Warunek;

DELETE FROM Ksiazki
   WHERE RokWydania < 1937 AND Cena < 123;

DELETE FROM Ksiazki
   WHERE Sygnatura IN (SELECT Sygnatura
                       FROM KsiazkiZagubione
                       WHERE StatusStraty = 'pokryta');
```

Modyfikacja rekordów

```
UPDATE   Tabela
   SET kolumna1 = nowa_wartosc,
       kolumna2 = nowa_wartosc,...,
       kolumnak = nowa_wartosc
   WHERE Warunek;

UPDATE  Czytelnicy
   SET Ulica = 'Krolewska',
       Limit =  15
   WHERE IdCzytelnika = 'PU1010';

UPDATE  Ksiazki
   SET Cena = Cena * 1.2,
   WHERE RokWydania > '1998';
```

Inne operacje

Projekcja na atrybuty `kolumna1, kolumna2,...,kolumnak`:

```
SELECT kolumna1, kolumna2,...,kolumnak
   FROM Tabela;
```

Selekcja dla warunku `KryteriumSelekcji`:

```
SELECT *
   FROM Tabela
   WHERE KryteriumSelekcji;
```

Iloczyn kartezjański:

```
SELECT TabelaA.*, TabelaB.*
   FROM TabelaA, TabelaB;
```

```
SELECT TabelaA.*, TabelaB.*
   FROM TabelaA CROSS JOIN TabelaB;
```

Różnica lewa (prawa):

```
SELECT TabelaA.*  (TabelaB.*)
   FROM TabelaA UNION JOIN TabelaB
   WHERE TabelaB.AtrybutB IS NULL;
  (WHERE TabelaA.AtrybutA IS NULL;)
```

Dopełnienie dla wybranego atrybutu z innej tabeli:

```
SELECT DISTINCT WybranyAtrybut
     FROM TabelaOdniesienia
     EXCEPT
     SELECT WybranyAtrybut
        FROM TabelaDana;
```

Studia podyplomowe ”Inżynieria oprogramowania” współfinansowane przez Unię Europejską w ramach Europejskiego Funduszu Społecznego

Projekt ”Studia podyplomowe z zakresu wytwarzania oprogramowania oraz zarządzania projektami w firmach informatycznych” realizowany w ramach Programu Operacyjnego Kapitał Ludzki