

Rozdział 7

Język manipulowania danymi DML

wstawianie danych i polecenie INSERT,
modyfikowanie danych i polecenie UPDATE,
usuwanie danych i polecenie DELETE, połączenia
modyfikowalne, sekwencje.



Wstawianie krotek do relacji (1)

- Wstawiając wartości do wszystkich atrybutów relacji można pominąć listę atrybutów. Wartości w klauzuli INSERT muszą występować w takiej samej kolejności, w jakiej występowały definicje atrybutów w poleceniu CREATE TABLE. W celu zwiększenia czytelności polecenia zaleca się podawanie listy nazw atrybutów.

```
INSERT INTO nazwa_relacji  
VALUES (wartość1 [ DEFAULT ] [ NULL ], ..., wartośćn);
```

```
INSERT INTO nazwa_relacji (atrybut1, ..., atrybutn)  
VALUES (wartość1, ..., wartośćn);
```

```
INSERT INTO zespoły VALUES (60, 'MULTIMEDIA', NULL);  
INSERT INTO zespoły (id_zesp, nazwa) VALUES (70, 'GRAFIKA');
```



Wstawianie krotek do relacji (2)

- Wstawianie parametryzowane w SQL*Plus oraz iSQL*Plus

```
INSERT INTO zespoły (id_zesp, nazwa, adres)  
VALUES (&identyfikator, '&nazwa', '&adres');
```

- Wstawianie krotek będących wynikiem zapytania

```
INSERT INTO nazwa_relacji [ (atrybut1, ..., atrybutn) ]  
SELECT (atrybut1, ..., atrybutn) FROM ... WHERE ... ;
```

```
INSERT INTO prac30 (numer_prac, nazwisko_prac, nazwa_zesp)  
SELECT id_prac, nazwisko, nazwa  
FROM pracownicy JOIN zespoły USING (id_zesp)  
WHERE id_zesp=30;
```



INSERT w standardzie SQL

- Wstawianie krotki wypełnionej wartościami domyślnymi

```
INSERT INTO nazwa_relacji  
DEFAULT VALUES;
```

- Wstawianie wielu krotek w jednym poleceniu

```
INSERT INTO nazwa_relacji (atrybut1, ..., atrybutn)  
VALUES (wartośća1, ..., wartośćan),  
(wartośćb1, ..., wartośćbn),  
(wartośćc1, ..., wartośćcn);
```



Modyfikowanie krotek relacji (1)

- Polecenie UPDATE

```
UPDATE relacja
SET  atrybut1 = wartość [ DEFAULT ] [ NULL ],
     atrybut2 = wartość [, ...]
[ WHERE warunek ];
```

Uwaga!

Pominięcie klauzuli WHERE spowoduje, że zmodyfikowane zostaną wszystkie krotki w relacji (warunek w klauzuli WHERE będzie spełniony dla wszystkich krotek).

```
UPDATE pracownicy
SET  etat = 'PROFESOR',
     placa_pod = placa_pod * 2.5
WHERE nazwisko = 'KOSZLAJDA';
```



Modyfikowanie krotek relacji (2)

- Inną wersją polecenia UPDATE jest polecenie wykorzystujące podzapytania skorelowane i/lub zagnieżdżone.

```
UPDATE relacja_A
SET atrybutA1 = (
    SELECT atrybutB1
    FROM relacja_B [ WHERE ... ] )
[ WHERE ... ];
```

```
UPDATE relacja_A
SET (atrybutA1, atrybutA2) = (
    SELECT atrybutB1, atrybutB2
    FROM relacja_B [ WHERE ... ] )
[ WHERE ... ];
```



Modyfikowanie krotek - przykłady

- Zwiększ płacę podstawową do wartości równej 120% średniej płacy podstawowej w zespole pracownika oraz zwiększ płacę dodatkową do wartości równej maksymalnej płacy dodatkowej w zespole pracownika. Operacji dokonaj tylko dla pracowników zatrudnionych po 1992 roku.

```
UPDATE pracownicy p
SET (p.placa_pod, p.placa_dod) =
    (SELECT 1.2 * AVG(placa_pod), MAX(placa_dod)
     FROM pracownicy WHERE id_zesp = p.id_zesp)
WHERE p.zatrudniony >= DATE '1993-01-01';
```

- Pracownikom posiadającym podwładnych zwiększ płacę dodatkową o 10% sumy płac podstawowych podwładnych.

```
UPDATE pracownicy s
SET s.placa_dod = s.placa_dod + 0.1 * (SELECT SUM(placa_pod)
                                     FROM pracownicy WHERE id_szefa = s.id_prac)
WHERE EXISTS
    (SELECT * FROM pracownicy WHERE id_szefa = s.id_prac);
```



Usuwanie krotek relacji

- Polecenie DELETE

```
DELETE [ FROM ] relacja
[ WHERE warunek ];
```

- Klauzula WHERE określa, które krotki należy usunąć z relacji. Jeżeli klauzula WHERE nie zostanie wyspecyfikowana, to usunięte zostaną wszystkie krotki z relacji.

```
DELETE FROM pracownicy
WHERE nazwisko IN ('BIAŁY', 'KONOPKA');
```

```
DELETE FROM pracownicy p
WHERE p.placa_pod <
    (SELECT AVG(placa_pod)
     FROM pracownicy
     WHERE id_zesp = p.id_zesp);
```



Modyfikowanie i usuwanie wyniku połączenia

- Jeśli w połączeniu dwóch relacji kolumna pochodzi z relacji zachowującej klucz (**key preserving table**), to taka kolumna może być modyfikowana.

UPDATE

```
(SELECT nazwa, nazwisko, etat, placa_pod
FROM pracownicy JOIN zespoły USING (id_zesp)
WHERE adres = 'PIOTROWO 3A')
SET placa_pod = 2000
WHERE etat = 'ASYSTENT';
```

DELETE FROM

```
(SELECT p.nazwisko AS pracownik, s.nazwisko AS szef
FROM pracownicy p JOIN pracownicy s
ON (p.id_szefa = s.id_prac))
WHERE szef = 'MORZY';
```



Sekwencje

- Sekwencja to obiekt bazy danych generujący kolejne liczby. Sekwencje są stosowane przede wszystkim do tworzenia kolejnych wartości sztucznych kluczy podstawowych.

- Tworzenie sekwencji

```
CREATE SEQUENCE nazwa_sekwencji [ START WITH n ]
[ INCREMENT BY m ]
[ MAXVALUE x | NOMAXVALUE ]
[ MINVALUE y | NOMINVALUE ];
```

- Modyfikowanie sekwencji

```
ALTER SEQUENCE nazwa_sekwencji [ INCREMENT BY m ]
[ MAXVALUE x | NOMAXVALUE ]
[ MINVALUE y | NOMINVALUE ];
```



Korzystanie z sekwencji

- Pseudokolumny NEXTVAL, CURRVAL

```
CREATE SEQUENCE myseq START WITH 300 INCREMENT BY 10;
SELECT myseq.NEXTVAL FROM DUAL;
SELECT myseq.CURRVAL FROM DUAL;
```

- Wykorzystanie sekwencji w poleceniach DML

```
INSERT INTO pracownicy (id_prac, nazwisko, etat)
VALUES (myseq.NEXTVAL, 'BOROWIAK', 'STAZYSTA');
```

```
UPDATE pracownicy
SET id_szefa = myseq.NEXTVAL
WHERE nazwisko = 'BOROWIAK';
```



Usuwanie sekwencji

- Usunięcie sekwencji

```
DROP SEQUENCE nazwa_sekwencji;
```

- Informacje o sekwencjach w słowniku bazy danych
 - USER_SEQUENCES, ALL_SEQUENCES

Uwagi:

- Sekwencja nie jest związana z konkretną relacją i może być wykorzystywana dla różnych atrybutów.
- Sekwencja jest odczytywana zawsze, nawet jeśli transakcja zostanie wycofana.
- Sekwencji nie można stosować w: podzapytaniach, zapytaniach z klauzulą DISTINCT, GROUP BY, ORDER BY, w klauzuli WHERE, w zapytaniach z operatorami zbiorowymi, w definicji perspektywy, w definicji wartości domyślnej DEFAULT

