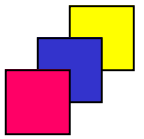
Język SQL. Rozdział 8.
Język manipulowania danymi DML

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



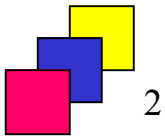
Wstawianie krotek do relacji (1)

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

Wstawianie krotek do relacji (2)

- **Wstawianie parametryzowane w narzędziach Oracle (SQL Developer oraz SQL*Plus):**

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

- **Wstawianie krotek będących wynikiem zapytania:**

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

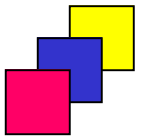
```
INSERT INTO prac30 (numer_prac, nazwisko_prac, nazwa_zesp)
        SELECT id_prac, nazwisko, nazwa
        FROM pracownicy JOIN zespoly USING (id_zesp)
        WHERE id_zesp=30;
```

Wstawianie krotek do relacji (3)

- **Dostarczenie wartości dla atrybutu przez podzapytanie:**

```
INSERT INTO pracownicy (id_prac, nazwisko, id_zesp)
VALUES(400, 'KOWALSKI', (SELECT id_zesp FROM zespoly
                         WHERE nazwa = 'ADMINISTRACJA'));
```

  - **podzapytanie umieszczamy w nawiasach,**
  - **podzapytanie musi zwrócić dokładnie jedną wartość.**



INSERT w standardzie SQL (1)

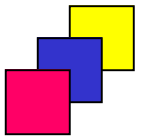
- **Wstawianie krotki wypełnionej wartościami domyślnymi:**

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

- **Wstawianie wielu krotek w jednym poleceniu:**

```
INSERT INTO nazwa_relacji (atrybut_1, ..., atrybut_n)
VALUES      (wartość_a_1, ..., wartość_a_n),
            (wartość_b_1, ..., wartość_b_n),
            (wartość_c_1, ..., wartość_c_n);
```

- **Uwaga! Powyższe polecenia nie są wspierane przez SZBD Oracle**



INSERT w standardzie SQL (2)

- **Alternatywa w Oracle:**

```
INSERT INTO nazwa_relacji (atrybut1, ..., atrybutn)
SELECT wartość_a1, ..., wartość_an FROM dual
UNION ALL
SELECT wartość_b1, ..., wartość_bn FROM dual
UNION ALL
…;
```

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 wyrażenieB1
        FROM relacja_B [WHERE ...] )
[WHERE ...];
```

```
UPDATE relacja_A
SET (atrybutA1, atrybutA2) = (
        SELECT wyrażenieB1, wyrażenieB2
        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 = nvl(s.placa_dod,0) + 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 (1)

- **Relacja zachowująca klucz (key preserving table) w wyniku połączenia dwóch relacji – jej klucz główny zachowuje swoje własności również w wyniku połączenia jej z inną relacją**

```
SELECT id_prac, nazwisko,
z.id_zesp, nazwa
FROM pracownicy p JOIN
zespoly z ON
p.id_zesp = z.id_zesp;
```

ID_PRAC	NAZWISKO	ID_ZESP	NAZWA
100	WEGLARZ	10	ADMINISTRACJA
110	BLAZEWICZ	40	ALGORYTMY
120	SLOWINSKI	30	SYSTEMY EKSPERCKIE
130	BRZEZINSKI	20	SYSTEMY ROZPROSZONE
140	MORZY	20	SYSTEMY ROZPROSZONE
150	KROLIKOWSK	20	SYSTEMY ROZPROSZONE
160	KOSZLAJDA	20	SYSTEMY ROZPROSZONE
170	JEZIERSKI	20	SYSTEMY ROZPROSZONE
190	MATYSIAK	20	SYSTEMY ROZPROSZONE
180	MAREK	10	ADMINISTRACJA
200	ZAKRZEWICZ	30	SYSTEMY EKSPERCKIE
210	BIALY	30	SYSTEMY EKSPERCKIE
220	KONOPKA	20	SYSTEMY ROZPROSZONE
230	HAPKE	30	SYSTEMY EKSPERCKIE

Klucz relacji *Pracownicy*

Klucz relacji *Zespoly*

Modyfikowanie i usuwanie wyniku połączenia (2)

- **Jeśli w połączeniu dwóch relacji kolumna pochodzi z relacji zachowującej klucz, to taka kolumna może być modyfikowana.**

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

```
DELETE FROM
        (SELECT etat
         FROM pracownicy NATURAL JOIN zespoly
         WHERE nazwa = ‘SYSTEMY ROZPROSZONE’)
WHERE etat = ‘PROFESOR';
```

Modyfikowanie i usuwanie wyniku połączenia (3)

```
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.**
- **Zastosowanie: do generacji kolejnych wartości sztucznych identyfikatorów numerycznych rekordów.**

- **Tworzenie sekwencji:**

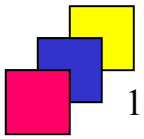
```
CREATE SEQUENCE nazwa_sekwencji [ START WITH n ]
  [INCREMENT BY m] [MAXVALUE x | NOMAXVALUE]
  [MINVALUE y | NOMINVALUE];
```

```
CREATE SEQUENCE myseq START WITH 300 INCREMENT BY 10;
```

- **Modyfikowanie istniejącej sekwencji:**

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

```
ALTER SEQUENCE myseq INCREMENT BY 5;
```

Korzystanie z sekwencji

- **Pseudokolumny sekwencji:**
  - ***NEXTVAL* – odczyt pseudokolumny powoduje generację kolejnego numeru z sekwencji, numer jest zwracany jako wartość pseudokolumny,**
  - ***CURRVAL* – jej wartość to ostatnio pobrany (przez użycie *NEXTVAL* lub *CURRVAL*) numer sekwencji; użycie nie generuje nowego numeru.**

- **Użycie sekwencji w poleceniach DML:**

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

UPDATE pracownicy SET id_szefa = myseq.CURRVAL
WHERE nazwisko = 'KONOPKA';
```

- **Usuwanie sekwencji:**

```
DROP SEQUENCE nazwa_sekwencji;
```

Dodatkowe informacje dot. sekwencji

- **Informacje o sekwencjach w słowniku bazy danych:**
  - **USER_SEQUENCES – sekwencje, będące własnością użytkownika,**
  - **ALL_SEQUENCES – sekwencje dostępne dla użytkownika.**

- **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 (do Oracle 11g włącznie).**