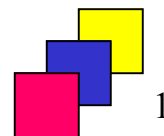


Język SQL. Rozdział 9.

Język definiowania danych DDL, część 1.

**Tworzenie relacji, typy danych, wartości domyślne
atrybutów, słownik bazy danych.**

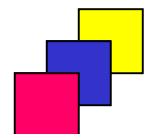


Tworzenie relacji

- polecenie CREATE TABLE

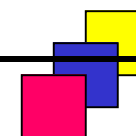
```
CREATE TABLE nazwa_relacji
    (nazwa_atrybutu typ (rozmiar) [DEFAULT wartość_domyślna]
      [ [CONSTRAINT nazwa_ogr] ograniczenie_atr],
      nazwa_atrybutu typ (rozmiar) [DEFAULT wartość_domyślna]
      [ [CONSTRAINT nazwa_ogr] ograniczenie_atr],
      ....
    [ [CONSTRAINT nazwa_ogr] ograniczenie_rel, ...] );
```

- Nazwa relacji:
 - musi zaczynać się od litery A-Za-z,
 - może zawierać litery, cyfry, znaki _ \$ # (ostatnie dwa nie są zalecane),
 - jest nieczuła na wielkość użytych znaków (chyba że użyto cudzysłowu),
 - nie może przekroczyć 30 znaków,
 - musi być jednoznaczna i różna od nazw innych relacji, perspektyw i synonimów w schemacie danego użytkownika,
 - nie może być słowem zastrzeżonym języka SQL.



Typy Oracle atrybutów relacji

Typ danych	Dopuszczalne wartości
CHAR(n BYTE CHAR)	Ciąg znaków o stałej długości i rozmiarze n bajtów lub znaków (dom. 1). Maks. 2000B
NCHAR(n)	Jak CHAR(n), n – liczba znaków, w UNICODE (2B lub 3B na znak)
VARCHAR2(n BYTE CHAR)	Ciąg znaków o zmiennej długości i rozmiarze n bajtów lub znaków. Maks. 4000B (32 767B od Oracle 12c) . Rozmiar n <u>musi</u> być podany.
NVARCHAR2(n)	Jak VARCHAR2(n), n – liczba znaków, w UNICODE (2B lub 3B na znak)
NUMBER(p,s)	Liczba o precyzji p (1-38) i skali s (-84,127) z przedziału 1×10^{-130} $9.9 \dots 9 \times 10^{125}$
DATE	Data z przedziału 1.01.4712 p.n.e. i 31.12.9999 n.e.
TIMESTAMP(p)	Znacznik czasowy z dokł. p części ułamkowych sekundy (od 0 do 9, domyślnie 6)
INTERVAL YEAR(p) TO MONTH	Przedział czasu reprezentowany przez lata i miesiące z zadaną liczbą cyfr w określeniu lat (od 0 do 9, domyślnie 6)
INTERVAL DAY(p1) TO SECOND(p2)	Przedział czasu reprezentowany przez dni, godziny, minuty i sekundy z zadaną liczbą cyfr w określeniu dni (p1) i liczbą pozycji ułamkowych części sekundy (p2)
CLOB	Duży obiekt binarny zawierający łańcuchy znaków (stałej i zmiennej długości) o maks. rozmiarze 128TB dla Oracle11g lub 8TB dla Oracle9i/10g
NCLOB	Jak CLOB, w UNICODE
BLOB	Duży obiekt binarny o maks. rozmiarze 128TB dla Oracle11g lub 8TB Oracle9i/10g
BFILE	Wskaźnik na plik systemu operacyjnego



Typ NUMBER (1)

- Zakres: $\langle 10^{-130}, 10^{126} - 1 \rangle$
- Zaimplementowany w sposób niezależny od platformy
- Deklaracja typu zmiennoprzecinkowego: NUMBER
 - maks. 40 pozycji

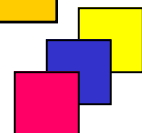
```
pr_placa NUMBER,
```

- Deklaracja typu stałoprzecinkowego: NUMBER

```
(precyzja, skala):
```

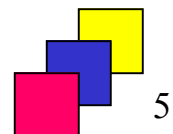
 - precyzja: $\langle 1, 38 \rangle$ - całkowita liczba pozycji znaczących
 - skala: $\langle -84, 127 \rangle$ - liczba pozycji na prawo (dodatnia) lub lewo (ujemna) od przecinka

```
pr_dodatek NUMBER(6,2),  
pr_wzrost NUMBER(3),  
pr_ułamek NUMBER(8,10),  
pr_tysiące NUMBER(1,-3),
```



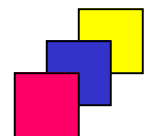
Typ NUMBER (2)

- **Typ stałoprzecinkowy:**
 - **przypadek 1. precyzja > skala, np. NUMBER(6,2):**
 - zakres: <-9 999,99, 9 999,99>
 - liczby zaokrąglane do dwóch miejsc po przecinku, np.:
 - 1 234,56 -> 1 234,56
 - 1 234,567 -> 1 234,57
 - > 9999,99 lub < -9999,99 -> błąd
 - **przypadek 2. skala = 0, np. NUMBER(4,0) lub NUMBER(4):**
 - liczba całkowita, zakres: <-9999, 9999>
 - liczby zaokrąglane do liczb całkowitych, np.:
 - 0,01 -> 0
 - 0,5 -> 1
 - 1234,5678 -> 1235
 - > 9999 lub < -9999 -> błąd



Typ NUMBER (3)

- **Typ stałoprzecinkowy (cd):**
 - **przypadek 3. precyzja < skala, np. NUMBER(8,10):**
 - zakres: <-0.0099999999, 0.0099999999>
 - liczby zaokrąglane do dziesięciu miejsc po przecinku, np.:
 - 0,0012345678 -> 0,0012345678
 - 0,00123456781 -> 0,0012345678
 - 0,00123456789 -> 0,0012345679
 - 0,00000000005 -> 0,0000000001
 - >=0,01 lub <= -0,01 -> błąd
 - **przypadek 4. skala < 0, np. NUMBER(1, -3):**
 - zakres: <-9000, 9000>
 - liczby całkowite zaokrąglane do najbliższego tysiąca, np.:
 - 499 -> 0
 - 500 -> 1 000
 - 9 499,99 -> 9 000
 - >=9500 lub <=-9500 -> błąd

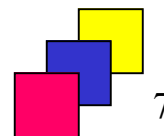


Podtypy typu NUMBER

- **DECIMAL**(precyzja, skala) – **NUMBER**(precyzja, skala),
- **FLOAT, DOUBLE PRECISION** – **NUMBER**(126), precyzja binarna,
- **FLOAT**(precyzja) – **NUMBER**(precyzja), precyzja binarna do 126 bitów,
- **INT, INTEGER, SMALLINT** – **NUMBER**(38),
- **NUMERIC**(precyzja, skala) – **NUMBER**(precyzja, skala),
- **REAL** – **FLOAT**(63).

Pozostałe typy numeryczne

- Typy zmiennoprzecinkowe z reprezentacją binarną wg standardu **IEEE-754**:
 - **BINARY_FLOAT** – 4 bajty,
 - **BINARY_DOUBLE** – 8 bajtów.



Ciągi znaków zmiennej długości (1)

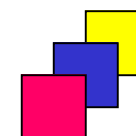
- **VARCHAR2** – ciąg znaków kodowany zestawem znaków domyślnym dla bazy, maks. długość 4000B (32 767B w Oracle12c),
 - długość podajemy w bajtach lub znakach:

```
nazwisko VARCHAR2(100 BYTE), imię VARCHAR2(50 CHAR),
```

- pominięcie **BYTE** i **CHAR** – długość wyrażona w jednostkach określonych przez parametr sesji **NLS_LENGTH_SEMANTICS**.

```
SELECT value FROM nls_session_parameters  
WHERE parameter = 'NLS_LENGTH_SEMANTICS';
```

- **NVARCHAR2** – ciąg znaków kodowany w wielobajtowym Unicode, może wykorzystywać inny zestaw znaków niż domyślny dla bazy.



Ciągi znaków zmiennej długości (2)

- Synonimy:
 - VARCHAR2: CHAR VARYING, CHARACTER VARYING, STRING, VARCHAR,
 - NVARCHAR2: NATIONAL CHAR VARYING, NCHAR VARYING, NATIONAL CHARACTER VARYING.
- Uwaga!

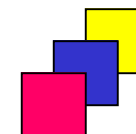
W Oracle pusty ciąg znaków jest **równy** NULL. Jest to **sprzeczne** ze standardem języka SQL.

```
SQL> create table TEST(tekst varchar2(10));
SQL> insert into TEST values(NULL);
SQL> insert into TEST values('');
SQL> select count(*) from TEST where tekst is null;
```

```
COUNT (*)
```

```
-----
```

```
2
```



Ciągi znaków stałej długości

- **CHAR** – ciąg znaków kodowany zestawem znaków domyślnym dla bazy, maks długość. 2000B (255B przed Oracle8i),
 - długość podajemy w bajtach lub znakach:

PESEL CHAR(11 BYTE), plec CHAR

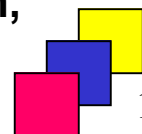
- pominięcie BYTE i CHAR – długość wyrażona w jednostkach określonych przez parametr sesji NLS_LENGTH_SEMANTICS,
 - pominięcie długości – długość ciągu = 1,
 - niewykorzystane pozycje dopełniane spacjami z prawej strony.
- **NCHAR** – ciąg znaków kodowany w wielobajtowym Unicode, może korzystać z innego zestawu znaków niż domyślny dla bazy
- **Synonimy:**
 - **CHAR: CHARACTER**
 - **NCHAR: NATIONAL CHARACTER, NATIONAL CHAR**

Kolumna IDENTITY (1)

- Kolumna, której wartości w rekordach będą generowane automatycznie (przy użyciu sekwencji).
- Najczęściej służy do identyfikowania rekordów (jako np. klucz podstawowy).
- Cechy:
 - tylko jedna kolumna IDENTITY w relacji,
 - kolumna domyślnie posiada ograniczenie NOT NULL,
 - kolumna nie może mieć zdefiniowanej klauzuli DEFAULT,
 - mechanizm dostępny od Oracle12c.
- Przykłady:

```
CREATE TABLE pracownicy  
(id_prac NUMBER(6) GENERATED ALWAYS AS IDENTITY,  
...)
```

- *id_prac* w kolejnych rekordach uzyska wartości od 1 z krokiem 1,
- podanie wartości dla *id_prac* przy wstawianiu rekordu zakończy się błędem,



Kolumna IDENTITY (2)

- Przykłady (cd):

```
CREATE TABLE pracownicy  
(id_prac NUMBER(6) GENERATED BY DEFAULT AS IDENTITY,  
...)
```

- id_prac* w kolejnych rekordach uzyska wartości od 1 z krokiem 1,
- podana przez użytkownika wartość dla *id_prac* przy wstawianiu rekordu zostanie uwzględniona,
- podanie wartości NULL zakończy się błędem,

```
CREATE TABLE pracownicy  
(id_prac NUMBER(6) GENERATED BY DEFAULT  
ON NULL AS IDENTITY, ...)
```

- j.w. + wartość zostanie wygenerowana automatycznie również wówczas, gdy przy wstawianiu dla *id_prac* podano wartość NULL.

Kolumna IDENTITY (3)

- Przykłady (cd):
 - *id_prac* w kolejnych rekordach uzyska wartości od 100 z krokiem 10.

```
CREATE TABLE pracownicy  
  (id_prac NUMBER(6) GENERATED ALWAYS  
   AS IDENTITY(START WITH 100 INCREMENT BY 10),  
  ...
```

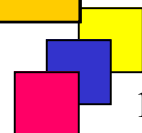
Wartości domyślne atrybutów (1)

- Każdemu atrybutowi można nadać domyślną wartość początkową. Robi się to za pomocą słowa kluczowego DEFAULT.

```
CREATE TABLE pracownicy (  
    id_prac NUMBER(6),  
    nazwisko VARCHAR2(50) DEFAULT 'NOWY PRACOWNIK',  
    data_zatrudnienia DATE DEFAULT SYSDATE,  
    pensja NUMBER(6,2) DEFAULT 1000,  
    badania_kontrolne DATE DEFAULT SYSDATE+365,  
    etat VARCHAR2(20) DEFAULT 'STAZYSTA');
```

- W Oracle12c wartością domyślną dla atrybutu może być sekwencja (alternatywa dla kolumny IDENTITY).

```
CREATE TABLE pracownicy (  
    id_prac NUMBER(6) DEFAULT seq_prac.nextval,  
    ...
```



Wartości domyślne atrybutów (2)

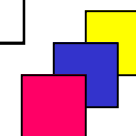
- **Działanie klauzuli DEFAULT podczas wstawiania danych:**

```
SQL> CREATE TABLE test(id NUMBER NOT NULL,  
                        tekst VARCHAR2(10) DEFAULT 'PUSTY' NOT NULL);  
  
SQL> INSERT INTO test(id) VALUES(10); -- OK  
SQL> INSERT INTO test(id, tekst) VALUES(20, NULL); -- Błąd
```

- **Od Oracle12c dostępna klauzula DEFAULT ON NULL:**

```
SQL> CREATE TABLE test(id NUMBER NOT NULL,  
                        tekst VARCHAR2(10) DEFAULT ON NULL 'PUSTY' NOT NULL);  
  
SQL> INSERT INTO test(id) VALUES(10); -- OK  
SQL> INSERT INTO test(id, tekst) VALUES(20, NULL); -- OK  
  
SQL> SELECT * FROM test;
```

ID	TEKST
10	PUSTY
20	PUSTY



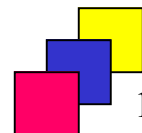
Tworzenie relacji przez podzapytanie

- Wynik zapytania można utrwalić w postaci relacji:
 - nowa relacja składa się z atrybutów wymienionych w klauzuli **SELECT** zapytania,
 - jeśli podano listę nazw atrybutów nowej relacji to lista atrybutów w klauzuli **SELECT** zapytania musi się pokrywać z tą listą.

```
CREATE TABLE nazwa_relacji  
    [ (nazwa_atrybutu [typ_wartości] [NULL | NOT NULL], ...) ]  
AS SELECT zapytanie;
```

```
CREATE TABLE roczne_place (nazwisko NOT NULL, etat, roczne)  
AS SELECT nazwisko, etat, 12 * placa_pod + NVL(placa_dod,0)  
FROM pracownicy;
```

```
CREATE TABLE pracownicy_zespoly AS  
SELECT nazwisko, nazwa, ROUND(SYSDATE-zatrudniony) AS dni  
FROM pracownicy JOIN zespoly USING (id_zesp);
```



Słownik bazy danych (dot. Oracle)

Perspektywa	Synonim	Opis
USER_OBJECTS	OBJ	Obiekty użytkownika
USER_TABLES	TABS	Relacje użytkownika
USER_TAB_COLUMNS	COLS	Atrybuty z relacji użytkownika

```
SELECT object_name, object_type FROM user_objects  
ORDER BY object_name;
```

```
SELECT table_name FROM user_tables ORDER BY table_name;
```

```
SELECT column_name FROM user_tab_columns  
WHERE table_name = 'PRACOWNICY'  
ORDER BY column_name;
```

