

## Rozdział 6

### Język definiowania danych DDL

Tworzenie relacji, typy danych, definiowanie atrybutów i ograniczeń integralnościowych, polecenie CREATE TABLE, wartości domyślne, modyfikowanie struktury relacji, zarządzanie ograniczeniami



## 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 atrybutów relacji

Typ danych	Dopuszczalne wartości
CHAR(n) NCHAR(n)	Ciąg znaków o stałej długości i rozmiarze n bajtów (domyślnie 1). Maksymalnie 2000B.
VARCHAR2(n) NVARCHAR2(n)	Ciąg znaków o zmiennej długości i rozmiarze n bajtów. Maksymalnie 4000B. Rozmiar n <u>musi</u> być podany.
NUMBER(p,s)	Liczba o precyzji p (1-38) i skali s (-84,127) z przedziału $1 \times 10^{-130}$ $9.9 \dots 9 \times 10^{125}$
DATE	Data z przedziału 1.01.4712 p.n.e. i 31.12.9999 n.e.
TIMESTAMP(p) INTERVAL	Czas liczony z dowolną dokładnością (Oracle 9i) p (0,9) Przedział czasu liczony z dowolną dokładnością (Oracle 9i)
LONG	Ciąg znaków o zmiennej długości i maksymalnym rozmiarze 2GB
RAW(n)	Ciąg bajtów o maksymalnym rozmiarze 2000B. Podanie rozmiaru n jest obowiązkowe.
LONG RAW	Ciąg bajtów o zmiennej długości i maksymalnym rozmiarze 2GB.
CLOB NCLOB	Duży obiekt binarny zawierający łańcuchy znaków (stałej i zmiennej długości) o maksymalnym rozmiarze 4GB.
BLOB	Duży obiekt binarny o maksymalnym rozmiarze 4GB.



## Typy ANSI

ANSI	ORACLE
CHARACTER(n)	CHAR(n)
CHARACTER VARYING(n) CHAR VARYING(n)	VARCHAR(n)
NATIONAL CHARACTER(n) NATIONAL CHAR(n)	NCHAR(n)
NATIONAL CHARACTER VARYING (n) NATIONAL CHAR VARYING(n) NCHAR VARYING(n)	NVARCHAR(n)
NUMERIC(p,s) DECIMAL(p,s)	NUMBER(p,s)
INTEGER, INT SMALLINT	NUMBER(38)
FLOAT(b) DOUBLE PRECISION REAL	NUMBER



## Nowe typy danych w Oracle9i

### TIMESTAMP(p)

przechowuje czas z dokładnością do p miejsc po przecinku  
(domyślnie 6, maksymalnie 9)

### INTERVAL YEAR(y) TO MONTH

przechowuje okres czasu liczony z dokładnością do miesięcy

### INTERVAL DAY(d) TO SECOND(s)

przechowuje okres czasu liczony z dokładnością do sekund

```
SELECT SYSTIMESTAMP FROM DUAL;

SELECT SYSDATE + INTERVAL '12-2' YEAR TO MONTH FROM DUAL;

SELECT SYSDATE + INTERVAL '1 1' DAY TO HOUR FROM DUAL;
```

## Ograniczenia integralnościowe atrybutu

atrybut typ(rozmiar) [ CONSTRAINT nazwa ] typ [warunek]

### Typ ograniczenia

NULL  
NOT NULL  
UNIQUE  
PRIMARY KEY  
REFERENCES  
  
ON DELETE CASCADE  
ON DELETE SET NULL  
  
CHECK

### Przykład

```
placa_dod NUMBER(6,2) NULL
placa_pod NUMBER(6,2) NOT NULL
nazwisko VARCHAR2(12) UNIQUE
numer NUMBER(4) PRIMARY KEY
id_zesp NUMBER(4)
    REFERENCES zespoly(id_zesp)
id_zesp NUMBER(4)
    REFERENCES zespoly(id_zesp)
    ON DELETE CASCADE
placa_pod NUMBER(6,2) CHECK
    (placa_pod BETWEEN 100 AND 3000)
```

## Ograniczenia integralnościowe relacji

[ CONSTRAINT nazwa ] typ (atrybut) [warunek]

### Typ ograniczenia

UNIQUE  
PRIMARY KEY  
FOREIGN KEY  
REFERENCES  
  
ON DELETE CASCADE  
ON DELETE SET NULL  
  
CHECK

### Przykład

```
CREATE TABLE pracownicy (
    id_prac ...,
    nazwisko ...,
    ...,
    UNIQUE (nazwisko),
    PRIMARY KEY (id_prac),
    CONSTRAINT p_fk
        FOREIGN KEY (id_zesp)
        REFERENCES zespoly (id_zesp)
        ON DELETE SET NULL );
```

## Wartości domyślne atrybutów

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

```
CREATE TABLE pracownicy (
    id_prac NUMBER(6) NOT NULL,
    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',
    ... );
```

## Tworzenie relacji – przykład (1)

Tabele *dydaktycy* i *przedmioty* przechowują odpowiednio dane wszystkich nauczycieli i dane o wykładanych przedmiotach.

```
CREATE TABLE dydaktycy (  
  id_dydaktyka NUMBER(2) CONSTRAINT id_dydaktyka_pk PRIMARY KEY,  
  nazwisko VARCHAR2(15) NOT NULL UNIQUE,  
  tytuł VARCHAR2(10) NOT NULL );  
  
CREATE TABLE przedmioty (  
  id_przedmiotu NUMBER(2) CONSTRAINT id_przedmiotu_pk PRIMARY KEY,  
  nazwa VARCHAR2(15) NOT NULL UNIQUE );
```

## Tworzenie relacji - przykład (2)

Tabela *zajecia* łączy dane z tabel *dydaktycy* i *przedmioty*, w tej tabeli przechowujemy dane o tym, kto wykłada jaki przedmiot i w jakiej formie.

```
CREATE TABLE zajecia (  
  id_zajec NUMBER(2) PRIMARY KEY,  
  rodzaj_zaj VARCHAR2(15)  
    CHECK (rodzaj_zaj IN ('wykład','ćwiczenia','laboratorium','projekt' ) ),  
  id_dydaktyka NUMBER(2),  
  id_przedmiotu NUMBER(2),  
  FOREIGN KEY (id_dydaktyka) REFERENCES dydaktycy(id_dydaktyka)  
    ON DELETE SET NULL,  
  FOREIGN KEY (id_przedmiotu) REFERENCES przedmioty(id_przedmiotu)  
    ON DELETE CASCADE );
```

## Tworzenie relacji przez podzapytanie

Wynik zapytania można zmaterializować 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 [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);
```

## Modyfikowanie schematu relacji

- Dodawanie nowych atrybutów i ograniczeń

```
ALTER TABLE nazwa_relacji  
ADD [ nazwa typ(rozmiar) [DEFAULT wartość] ograniczenia |  
  CONSTRAINT nazwa typ ograniczenie ];
```

- Modyfikowanie istniejących atrybutów

```
ALTER TABLE nazwa_relacji  
MODIFY ( nazwa typ(rozmiar) [DEFAULT wartość] [ NOT NULL ] );
```

- Usuwanie atrybutów i ograniczeń

```
ALTER TABLE nazwa_relacji  
DROP [ COLUMN ( nazwa ) | CONSTRAINT ( nazwa ) ];
```

## Zarządzanie ograniczeniami integralnościowymi

- Włączenie ograniczenia integralnościowego

```
ALTER TABLE relacja  
ENABLE [CONSTRAINT nazwa | rodzaj]
```

```
ALTER TABLE pracownicy  
ENABLE CONSTRAINT prac_fk;
```

- Wyłączenie ograniczenia integralnościowego

```
ALTER TABLE relacja  
DISABLE [CONSTRAINT nazwa | rodzaj]
```

```
ALTER TABLE pracownicy  
DISABLE PRIMARY KEY;
```

## Zmiana nazwy relacji, usuwanie relacji

- Zmiana nazwy istniejącej relacji

```
RENAME stara_nazwa TO nowa_nazwa;
```

- Dodanie komentarza do relacji

```
COMMENT ON TABLE relacja IS 'komentarz';  
COMMENT ON COLUMN relacja.atrybut IS 'komentarz';
```

- Usunięcie relacji

```
DROP TABLE nazwa_relacji [CASCADE CONSTRAINTS];
```

- wszystkie dane są usuwane z relacji
- wszystkie indeksy założone na relacji są usuwane
- jeżeli nie podano CASCADE CONSTRAINTS to polecenie może zakończyć się błędem (jeśli istnieją relacje zależne)

## Słownik bazy danych

- Klasy perspektyw słownikowych
  - USER\_xxx, ALL\_xxx, DBA\_xxx

Perspektywa	Synonim	Opis
DICTIONARY	DICT	Wszystkie obiekty b.d.
USER_OBJECTS	OBJ	Obiekty użytkownika
USER_TABLES	TABS	Relacje użytkownika
USER_TAB_COLUMNS	COLS	Atrybuty z relacji i perspektyw użyt.
USER_COL_COMMENTS		Komentarze dla atrybutów
USER_TAB_COMMENTS		Komentarze dla relacji
USER_CONSTRAINTS		Ograniczenia integralnościowe
USER_CONS_COLUMNS		Atrybuty z ograniczeń integraln.