Bazy danych

Wykład 4_3

Temat: Deklaratywne więzy integralności w języku T-SQL

Sławomir Świętoniowski

slawomir-swietoniowski@wp.pl

Plan wykładu

1. **Integralność relacyjnej bazy danych.**
2. Więzy integralności w MS SQL Server 2000.

Więzy integralności w bazie danych

- Baza danych jest w stanie **integralności** (spójności), jeśli dane w niej przechowywane są prawidłowe, to znaczy zgodnie z przyjętymi założeniami nie zawierają błędów.
- W transakcyjnych bazach danych zachowanie integralności jest wymogiem absolutnie podstawowym, nadrzędnym wobec wydajności.
- Ze względu na sposób realizacji, rozróżniamy dwa rodzaje integralności:
  - **integralność wewnętrzna** – więzy definiowane na poziomie tabel (szybkie i zalecane, ale posiadające ograniczone możliwości).
  - **integralność dodatkowa** – więzy definiowane zewnętrznie, czyli poza tabelami (duże możliwości, ale zazwyczaj wolniejsze działanie) za pomocą procedur przechowywanych i wyzwalaczy.

Rodzaje więzów integralności

Ze względu na cel stosowania, reguły integralności w bazie danych można podzielić na cztery kategorie.

- **Integralność encji** (spójność jednostkowa) – zapewnia jednoznaczność odwołania do wierszy zapisanych w tabeli, ponieważ każdy wierszy musi się różnić od pozostałych. Własność tę zapewnia klucz główny tabeli (PRIMARY KEY).
- **Integralność dziedziny** – sprawia, że w danej kolumnie wprowadzane są tylko wartości dopuszczalne, zgodne z założeniami systemu. Jest ona realizowana przez definicję typu danych pola, własności NULL / NOT NULL, ograniczenia CHECK, DEFAULT, FOREIGN KEY oraz reguły.
- **Integralność referencyjna** (spójność odwoławcza) – gwarantuje, że jeżeli w danej tabeli jest odwołanie do wierszy w innej tabeli, wiersze te rzeczywiście tam istnieją, a relacja jest zgodna z założeniami systemu.
- **Integralność zdefiniowana przez użytkownika** – reguły, ograniczenia, zasady działania systemu, które nie należą do żadnej z powyższych kategorii. Najczęściej są to więzy skomplikowane, realizowane proceduralnie w samej bazie, albo w innych warstwach aplikacji.

Klucze główne i obce

- **Klucz główny** (ang. *primary key*, PK):
  - zbiór atrybutów, który identyfikuje jednoznacznie wiersze tabeli;
  - w tabeli może być kilka kluczy kandydujących, spośród których wybieramy jeden klucz główny (np. w tabeli [Osoba] kluczami kandydującymi mogą być kolumny [NIP], [PESEL], [IdOsoby]).
- **Klucz obcy** (ang. *foreign key*, FK):
  - pozwala na łączenie danych z różnych tabel;
  - zbiór atrybutów w tabeli, który czerpie swoje wartości z tej samej dziedziny, co klucz główny tabeli powiązanej.

Integralność encji

- Każda tabela musi mieć klucz główny, który jednoznacznie identyfikuje wiersze tej tabeli.
- Klucz główny nie może zawierać wartości pustych (NULL).
- Zabronione są powtórzenia wierszy w ramach jednej tabeli.

Integralność referencyjna

- Klucz obcy może przyjmować jedną z dwóch wartości:
  - klucz główny z tabeli powiązanej;
  - wartość NULL (jeżeli nie koliduje to z innymi regułami integralności).
- Niedozwolone są wskazania poprzez klucz obcy na wiersz, który nie istnieje.
- Kaskadowa aktualizacja i usuwanie zależą od konkretnego zastosowania (np. jeśli usuwamy fakturę VAT z tabeli [Faktura], to usuwamy także wszystkie jej pozycje z powiązanej tabeli [Pozycja]; natomiast jeśli usuwamy grupę studentów z tabeli [Grupa], to raczej nie usuwamy z bazy wszystkich studentów z tej grupy, zapisanych w powiązanej tabeli [Student]).

Plan wykładu

1. Integralność relacyjnej bazy danych.

2. Więzy integralności w MS SQL Server 2008 R2.

Więzy integralności w MS SQL Server

- **Więzy deklaratywne** (preferowane) - więzy statyczne, stosunkowo łatwe do tworzenia i zarządzania, nie obciążają zbytnio systemu**:**
  - ograniczenia (ang. *constraints*);
  - reguły (ang. *rules*);
  - wartości domyślne (ang. *default values*).
- **Metody proceduralne** (wspomagające) - więzy dynamiczne, pozwalające na oprogramowanie skomplikowanych zależności i reguł biznesowych; obciążają system bardziej, niż deklaracje**:**
  - procedury przechowywane (ang. *stored procedures*);
  - procedury wyzwalane, wyzwalacze (ang. *triggers*);
  - *kod innych warstw aplikacji* (poza warstwą bazy danych).

Ograniczenia (Constraints)

Ograniczenia są podstawowym środkiem zachowania spójności danych.
Rodzaje ograniczeń:

- PRIMARY KEY – klucz główny;
- FOREIGN KEY – klucz obcy;
- UNIQUE – pole bez powtórzeń wartości;
- CHECK – sprawdzenie warunku;
- DEFAULT – wartość domyślna pola.

Aktualny stan ograniczeń w tabeli możemy sprawdzać za pomocą procedury systemowej „**sp_helpconstraint**” (np. sp_helpconstraint Student).

PRIMARY KEY – klucz główny

- Ograniczenie PRIMARY KEY służy do zachowania integralności encji poprzez definiowanie klucza głównego tabeli (bez powtórzeń wartości i bez wartości NULL).
- Klucz powinien być jak najkrótszy (najlepiej pojedyncza kolumna). Jeżeli konieczne jest tworzenie klucza z wielu kolumn, należy rozważyć dodanie klucza zastępczego (Id_kolumny).
- Domyślnie na kolumnie (kolumnach) klucza głównego PRIMARY KEY system DBMS tworzy indeks zgrupowany (CLUSTERED INDEX), który przyśpiesza wyszukiwanie wierszy w tabeli.

PRIMARY KEY – przykład

```
-- Definicja tabeli bez klucza.
CREATE TABLE Przedmiot
(
    KodPrzedmiotu char (7) NOT NULL,
    Nazwa char (128) NOT NULL,
    Semestr tinyint NOT NULL,
    ECTS tinyint NOT NULL,
    FormaZajec char (10) NOT NULL,
    Opiekun int NULL
)
GO
-- Oddzielna definicja klucza głównego – rozwiązanie zwiększające
-- elastyczność skryptu T-SQL.
ALTER TABLE Przedmiot ADD
    CONSTRAINT PK_Przedmiot PRIMARY KEY CLUSTERED (KodPrzedmiotu)
GO
```

FOREIGN KEY – klucz obcy

- Ograniczenie FOREIGN KEY zapewnia integralność referencyjną i umożliwia definiowanie schematu relacyjnego (powiązań pomiędzy tabelami).
- Kolumna z ograniczeniem FOREIGN KEY wskazuje na kolumnę w innej tabeli lub inną kolumnę w tej samej tabeli. Obie kolumny muszą być tego samego typu.
- Kolumna, na którą wskazuje FOREIGN KEY musi mieć ograniczenie PRIMARY KEY lub UNIQUE.

FOREIGN KEY – przykłady (1)

- Definiowanie klucza obcego w poleceniu CREATE TABLE.

```
CREATE TABLE Przedmiot
(
    KodPrzedmiotu char (7) NOT NULL,
    Nazwa char (128) NOT NULL,
    Semestr tinyint NOT NULL,
    ECTS tinyint NOT NULL,
    FormaZajec char (10) NOT NULL,
    Opiekun int CONSTRAINT FK_Przedmiot_Opiekun
    REFERENCES Pracownik (NrPracownika) NULL
)
GO
```

FOREIGN KEY – przykłady (2)

- Dodawanie klucza obcego za pomocą polecenia ALTER TABLE.

```
-- Definicja tabeli bez klucza głównego i obcego.
CREATE TABLE Przedmiot
(
    KodPrzedmiotu char (7) NOT NULL,
    Nazwa char (128) NOT NULL,
    Semestr tinyint NOT NULL,
    ECTS tinyint NOT NULL,
    FormaZajec char (10) NOT NULL,
    Opiekun int NULL
)
GO
-- Oddzielna definicja klucza obcego – rozwiązanie zwiększające elastyczność skryptu.
ALTER TABLE Przedmiot ADD
    CONSTRAINT FK_Przedmiot_Opiekun FOREIGN KEY (Opiekun)
    REFERENCES Pracownik (NrPracownika)
GO
```

Przekazywanie integralności referencyjnej

- Służy do wymuszenie kaskadowego aktualizowania i kasowania wierszy w tabeli powiązanej (wskazywanej przez FOREIGN KEY) przy odpowiedniej zmianie wiersza w tabeli macierzystej.
- Do przekazywania integralności służą klauzule:
  - ON UPDATE - przy aktualizacji;
  - ON DELETE - przy kasowaniu.
- Do powyższych klauzul dodajemy słowa:
  - CASCADE - wymusza kaskadowe aktualizowanie lub kasowanie;
  - NO ACTION - wyłącza przekazywanie zmian.
- Należy dobrze przemyśleć użycie tych opcji, zgodnie z zależnościami w dziedzinie, którą modelujemy (np. jeżeli w hurtowni usuwamy fakturę VAT, to powinniśmy usunąć także wszystkie jej pozycje; natomiast jeżeli w szkole usuwamy grupę (np. z powodu zbyt małej liczebności), to niekoniecznie usuwamy automatycznie wszystkich studentów, którzy do niej należeli).

Przekazywanie integralności referencyjnej – przykład

- Definiowanie przekazywania zmian za pomocą ALTER TABLE.

```
ALTER TABLE Oplata
ADD CONSTRAINT FK_Oplata_Student FOREIGN KEY (NrIndeksu)
REFERENCES Student (NrIndeksu)
    ON UPDATE CASCADE
    ON DELETE CASCADE
GO
```

UNIQUE – pole bez powtórzeń

- Ograniczenie UNIQUE jest podobne do PRIMARY KEY, lecz dopuszcza wystąpienie jednej wartości NULL. Zaleca się jednak ustawienie NOT NULL (ponieważ powyższe zachowanie jest nieprzewidywalne).
- UNIQUE stosujemy wówczas, gdy wymagamy, aby dane pole nie zawierało powtarzających się wartości (np. NIP, Nr_faktury).
- Nałożenie ograniczenia UNIQUE na określone kolumny powoduje utworzenie na nich unikalnego indeksu niezgrupowanego (unique, nonclustered index).

UNIQUE – przykłady

```
-- Kolumny [NIP] i [PESEL] nie mogą zawierać powtórzonych wartości.
ALTER TABLE Pracownik ADD
CONSTRAINT UN_Pracownik_NIP UNIQUE (NIP)
GO

ALTER TABLE Pracownik ADD
CONSTRAINT UN_Pracownik_PESEL UNIQUE (PESEL)
GO
```

CHECK – sprawdzenie warunku

- Ograniczenie CHECK określa, jakie wartości są dopuszczalne w danej kolumnie.
- Ograniczenie CHECK jest realizowane za pomocą wyrażenia logicznego, które musi przyjąć wartość TRUE, aby była możliwa modyfikacja danych w określonym polu.
- Wyrażenie sprawdzane w ograniczeniu CHECK może się odnosić do kolumn w tej samej tabeli albo do funkcji bez parametrów. Nie można odwoływać się do pól w innych tabelach.

CHECK – przykłady

```
-- Wymuszenie masek wprowadzania dla numeru PESEL i kodu pocztowego.
ALTER TABLE Pracownik ADD
CONSTRAINT CK_Pracownik_PESEL CHECK
    (PESEL LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'),
CONSTRAINT CK_Pracownik_KodPocztowy CHECK
    (KodPocztowy LIKE '[0-9][0-9]-[0-9][0-9][0-9]')
GO

-- Dopuszczalny zakres numeru semestru na studiach od 1 do 7 włącznie.
ALTER TABLE Przedmiot ADD
CONSTRAINT CK_Przedmiot_Semestr CHECK (Semestr BETWEEN 1 AND 7)
GO
```

Wartości domyślne – DEFAULTS

- Wartości domyślne określają, jaką wartość ma przyjmować dana kolumna w tabeli, jeżeli nie zostanie jawnie wprowadzona.
- Wartościami domyślymi mogą być: stałe, wbudowane funkcje oraz wyrażenia arytmetyczne.
- Rodzaje wartości domyślnych (identyczne pod względem funkcjonalności, ale różnie implementowane):
  - **deklaratywne** – definiowane jako ograniczenie DEFAULT w tabeli,
  - **związywane** – tworzone jako oddzielne obiekty w bazie danych.
- Zastosowanie wartości domyślnych:
  - wypełnianie kolumn (np. wartością „nieznany”), które normalnie miałyby wartość NULL;
  - wprowadzanie wartości najczęściej używanych (np. domyślna miejscowość zamieszkania pracownika – „Wrocław”);
  - generowanie danych przez funkcje systemowe (np. GETDATE( )).

Deklaratywne wartości domyślne

Jako rodzaj ograniczeń są implementowane za pomocą poleceń CREATE TABLE oraz ALTER TABLE.

```
ALTER TABLE Student ADD
-- Domyślna wartość numeru indeksu.
CONSTRAINT DF_Student_NrIndeksu DEFAULT (1) FOR NrIndeksu,
-- Data systemowa jako wartość domyślna pola.
CONSTRAINT DF_Student_DataZapisania DEFAULT (GETDATE( ))
FOR DataZapisania,
-- Ustalona data wskazująca na niewykorzystane pole (zamiast NULL).
CONSTRAINT DF_Student_DataWypisania DEFAULT ('01-01-1900')
FOR DataWypisania
GO

-- Usunięcie ograniczenia DEFAULT.
ALTER TABLE Student DROP CONSTRAINT DF_Student_NrIndeksu
GO
```

Literatura

1. BEYNON-DAVIES P., *Systemy baz danych – nowe wydanie*, WNT, Warszawa 2003.
2. DATE C. J., DARWEN H., *SQL. Omówienie standardu języka*, WNT, Warszawa 2000 (książka dostępna w bibliotece WSIZ "Copernicus").
3. MICROSOFT, *Books On-Line* – dokumentacja systemu *MS SQL Server*, Microsoft Corp. 1988 – 2000.
4. RANKINS R., JENSEN P., BERTUCCI P., *Microsoft SQL Server 2000. Księga eksperta.*, HELION, Gliwice 2003 (książka dostępna w bibliotece WSIZ „Copernicus").
5. Strona MSDN: http://msdn.microsoft.com.
6. WAYMIRE R., SAWTELL R., *MS SQL Server 2000 dla każdego*, HELION, Gliwice 2002 (książka dostępna w bibliotece WSIZ "Copernicus").

Bazy danych

Wykład 4_3

Dziękuję za uwagę !