Базы данных

Тема 7

СУБД SQL Server 2012 – Категории целостности данных.

Категории целостности данных

- целостность данных подразделяется на следующие категории:
  - *сущностная целостность*: определяет строку как уникальную сущность в конкретной таблице;
  - *доменная целостность*: определяет достоверность записей в конкретном столбце;
  - *ссылочная целостность*: сохраняет определенные связи между таблицами при вводе или удалении записей;
  - *пользовательская целостность*: позволяет определять бизнес-правила, не входящие ни в одну из категорий целостности.

Ссылочная целостность

- в SQL Server 2012 ссылочная целостность основана на связи первичных и внешних ключей (либо внешних и уникальных ключей) и обеспечивается с помощью ограничений FOREIGN KEY и CHECK;
- ссылочная целостность гарантирует согласованность значений ключей во всех таблицах;
- ссылочная целостность требует отсутствия ссылок на несуществующие значения, а также обеспечивает согласованное изменение ссылок во всей базе данных при изменении значения ключа;
- при обеспечении ссылочной целостности SQL Server не допускает следующих действий пользователей:
  - добавления или изменения записей в связанной таблице, если в первичной таблице нет соответствующей записи;
  - изменения значений в первичной таблице, которое приводит к появлению потерянных записей в связанной таблице;
  - удаления записей из первичной таблицы, если имеются совпадающие ключи в других таблицах.

Сущностная, доменная и пользовательская целостность

- сущностная целостность обеспечивает целостность столбцов идентификаторов или первичного ключа таблицы с помощью:
  - индексов;
  - ограничений UNIQUE;
  - ограничений PRIMARY KEY;
  - свойств IDENTITY;
- доменная целостность включает:
  - ограничения типа данных;
  - ограничения формата при помощи ограничений CHECK;
  - ограничения диапазона возможных значений при помощи ограничений FOREIGN KEY, CHECK, DEFAULT, определений NOT NULL;
- поддержку пользовательской целостности обеспечивают все остальные категории целостности: любые типы ограничений уровня столбца и уровня таблицы в:
  - инструкции CREATE TABLE;
  - хранимых процедурах;
  - триггерах.

Типы данных

- с объектом, содержащим данные, связан тип, определяющий виды данных, которые могут храниться в объекте;
- типы данных имеют следующие объекты:
  - столбцы таблиц и представлений;
  - параметры хранимых процедур / функций;
  - переменные;
  - функции Transact-SQL, возвращающие одно или несколько значений;
  - хранимые процедуры, возвращающие значение (всегда имеет тип **integer**);
- можно создавать два вида пользовательских типов данных:
  - типы данных псевдонима - создаются на основе базовых типов данных (позволяют назначить типу данных имя, лучше характеризующее типы значений, которые будут храниться в объекте):

    CREATE TYPE birthday FROM datetime NULL
  - пользовательские типы данных CLR основаны на типах данных, созданных в управляемом коде и помещенных в сборку SQL Server.

Атрибуты типов данных

- при назначении типа данных объекту определяются четыре атрибута объекта:
  - вид данных, содержащихся в объекте;
  - размер или длина хранимого объектом значения: количество байт, используемых для хранения числа;
  - точность числа (только в случае численных типов): количество десятичных знаков в числе;
  - масштаб числа (только в случае численных типов): количество десятичных знаков справа от десятичного разделителя;
- точность (precision) и масштаб (scale) числовых типов данных, кроме **decimal** и **numeric**, фиксированы.

Типы данных SQL Server 2012

- битовый: **bit**;
- целые: **tinyint**, **smallint, int, bigint**;
- вещественные: **decimal**, **numeric**, **float**, **real**;
- финансовые: **smallmoney**, **money**;
- дата и время: **smalldatetime**, **datetime**, **datetime2**, **date**, **time**, **datetimeoffset**;
- строковые:
  - Unicode: **nchar**, **nvarchar**, **ntext**;
  - non-Unicode: **char**, **varchar**, **text**;
- двоичные: **binary**, **varbinary**, **image**;
- пространственные: **geography**, **geometry**;
- специальные: **cursor**, **table**, **uniqueidentifier**, **rowversion** (**timestamp**), **hierarchyid**, **sql_variant**, **xml**.

Преобразование типов данных в SQL Server 2012

- тип выражения определяется типом входящих в него операндов;
- если, например, оператор соединяет два выражения разных типов, то производится приведение значений левой и правой частей к типу с высшим приоритетом согласно правилам предшествования типов (*data type precedence*) – по мере убывания приоритета: пользовательские типы, **sql_variant, xml, datetimeoffset, datetime2, datetime, smalldatetime, date, time, float, real, decimal, money, smallmoney, bigint, int, smallint, tinyint, bit, ntext, text, image, timestamp, uniqueidentifier, nvarchar** (включая nvarchar(max)), **nchar, varchar** (включая varchar(max)), **char, varbinary** (включая varbinary(max)), **binary**;
- если неявное приведение типов невозможно, выдается сообщение об ошибке;
- явное приведение типов:
  - CAST (expression AS data_type [(length)])
  - CONVERT (data_type [(length)] , expression [, style])
- очевидно, что в ряде случаев может быть невозможно как неявное, так и явное приведение типов.

Ограничения и значения по умолчанию

- ограничения позволяют задать метод, с помощью которого компонент SQL Server 2012 Database Engine автоматически обеспечивает целостность базы данных:
  - NOT NULL указывает, что в столбце недопустимы значения NULL;
  - CHECK обеспечивают целостность домена путем ограничения значений, которые могут быть помещены в столбец;
  - UNIQUE обеспечивают уникальность значений в наборе столбцов (допускает значения NULL);
  - PRIMARY KEY используются для указания столбца или набора столбцов, которые имеют значения, уникально идентифицирующие строку в таблице (не допускает значения NULL);
  - FOREIGN KEY задают и обеспечивают связи между таблицами;
- ограничения могут относиться к столбцам или к таблицам;
- значения по умолчанию (DEFAULT) определяют, какими значениями заполнять столбец, если при вставке строки для этого столбца значение не указано. Значение по умолчанию могут быть любым выражением, результат которого – константа.

Создание, изменение и удаление таблиц

```
CREATE TABLE Orders (
   OrderID int IDENTITY(1, 1) NOT NULL,
   CustomerID nchar(5)
      COLLATE Cyrillic_General_CI_AS NULL,
   OrderDate datetime NOT NULL,
   Freight money NULL
      CONSTRAINT DF_Orders_Freight DEFAULT (0),
   ShipAddress nvarchar(60)
      COLLATE Cyrillic_General_CI_AS NULL,
   CONSTRAINT CHK_OrderDate
      CHECK (OrderDate > CONVERT(DATETIME, '1/1/1990', 103)),
   CONSTRAINT PK_Orders PRIMARY KEY CLUSTERED (OrderID),
   CONSTRAINT FK_Orders_Customers
      FOREIGN KEY (CustomerID) REFERENCES Customers (CustomerID)
)
GO

ALTER TABLE doc_exc
ADD column_b VARCHAR(20) NULL
CONSTRAINT exb_unique UNIQUE ;
GO

DROP TABLE Orders
GO
```

Создание и изменение столбцов идентификаторов: свойство IDENTITY

- свойства IDENTITY позволяет разработчику указать как начальное значение идентификатора для первой строки, вставляемой в таблицу, так и шаг его увеличения;
- при вставке значений в таблицу со столбцом идентификаторов SQL Server 2012 Database Engine автоматически формирует следующее значение идентификатора, добавляя значение шага приращения идентификатора к начальному значению;
- использование свойства IDENTITY:
  - уникальность свойства IDENTITY гарантирована только в рамках таблицы, в которой оно использовано;
  - таблица может содержать только один столбец со свойством IDENTITY;
  - столбец должен иметь тип данных **decimal**, **int**, **numeric**, **smallint**, **bigint** или **tinyint**;
  - можно указать начальное значение и шаг (по умолчанию, (1,1));
  - столбец идентификатора не должен допускать значений NULL и содержать определений или объектов по умолчанию;
  - ссылку на столбец в списке выборки можно создать с помощью указания ключевого слова $IDENTITY после свойства IDENTITY, также сослаться на столбец можно при помощи имени.

Глобальные уникальные идентификаторы

- несмотря на то, что свойство IDENTITY автоматизирует нумерацию строк в рамках одной таблицы, разные таблицы, каждая со своим столбцом идентификаторов, могут создавать одинаковые значения (уникальность свойства IDENTITY гарантирована только в рамках таблицы, в которой оно использовано);
- если в приложении нужно сформировать столбец идентификатора, уникальный для всей базы данных или всех баз данных во всех компьютерных сетях, необходимо использовать свойство ROWGUIDCOL, тип данных **uniqueidentifier** и функцию NEWID;
- использование свойства ROWGUIDCOL для определения столбца идентификаторов GUID:
  - в таблице может содержаться только один столбец ROWGUIDCOL, который должен иметь тип данных **uniqueidentifier**;
  - Database Engine не формирует значения для этого столбца автоматически;
  - не обеспечивается уникальность значений, хранимых в столбце;
  - для вставки глобального уникального значения, необходимо:
    - использовать для столбца определение DEFAULT, которое использует функцию NEWID для формирования глобального уникального значения;
    - использовать функцию NEWID в инструкции INSERT;
  - ссылку на столбец в списке выборки можно создать с помощью указания ключевого слова $ROWGUID после свойства ROWGUIDCOL.

Использование столбцов глобальных идентификаторов

```
CREATE TABLE [dbo].[PurchaseOrderDetail]
    (
    [PurchaseOrderID] [int] NOT NULL REFERENCES
          Purchasing.PurchaseOrderHeader(PurchaseOrderID),
    [LineNumber] [smallint] NOT NULL,
    [ProductID] [int] NULL REFERENCES Production.Product(ProductID),
    [UnitPrice] [money] NULL,
    [DueDate] [datetime] NULL,
    [rowguid] [uniqueidentifier] ROWGUIDCOL NOT NULL
          CONSTRAINT [DF_PurchaseOrderDetail_rowguid] DEFAULT (newid()),
    [ModifiedDate] [datetime] NOT NULL
          CONSTRAINT [DF_PurchaseOrderDetail_ModifiedDate] DEFAULT (getdate()),
    [LineTotal] AS (([UnitPrice]*[OrderQty])),
    [StockedQty] AS (([ReceivedQty]-[RejectedQty])),
          CONSTRAINT [PK_PurchaseOrderDetail_PurchaseOrderID_LineNumber]
          PRIMARY KEY
)
ON [PRIMARY]

CREATE TABLE MyUniqueTable
    (UniqueColumn UNIQUEIDENTIFIER DEFAULT NEWID(),
    Characters VARCHAR(10) )
GO

INSERT INTO MyUniqueTable(Characters) VALUES ('abc')
INSERT INTO MyUniqueTable VALUES (NEWID(), 'def')
GO
```

Последовательности

```
CREATE SEQUENCE [schema_name . ] sequence_name
    [ AS [ built_in_integer_type | user-
      defined_integer_type ] ]
    [ START WITH <constant> ]
    [ INCREMENT BY <constant> ]
    [ { MINVALUE [ <constant> ] } | { NO MINVALUE } ]
    [ { MAXVALUE [ <constant> ] } | { NO MAXVALUE } ]
    [ CYCLE | { NO CYCLE } ]
    [ { CACHE [ <constant> ] } | { NO CACHE } ]
    [ ; ]
```

- в отличие от полей IDENTITY:
  - очередное значение последовательности можно получить с помощью функции NEXT VALUE FOR;
  - последовательность не привязана к конкретной таблице, следовательно, возможно обеспечить уникальность значений более чем в одной таблице;
  - для полей, сформированных с помощью последовательностей, не контролируются неизменность и уникальность значений.

Использование последовательностей

```
--Create a table
CREATE TABLE Test.Orders
   (OrderID int PRIMARY KEY,
   Name varchar(20) NOT NULL,
   Qty int NOT NULL
);
GO

-- Create a sequence
CREATE SEQUENCE Test.CountBy1
   START WITH 1
   INCREMENT BY 1;
GO

-- Insert three records
INSERT Test.Orders (OrderID, Name, Qty)
VALUES (NEXT VALUE FOR Test.CountBy1, 'Tire', 2);
INSERT test.Orders (OrderID, Name, Qty)
VALUES (NEXT VALUE FOR Test.CountBy1, 'Seat', 1) ;
INSERT test.Orders (OrderID, Name, Qty)
VALUES (NEXT VALUE FOR Test.CountBy1, 'Brake', 1) ; GO
```

Каскадные ограничения ссылочной целостности

- с помощью каскадных ограничений ссылочной целостности можно определять действия, которые SQL Server 2012 будет предпринимать, когда пользователь попытается удалить или обновить ключ, на который указывают существующие внешние ключи;
- предложения REFERENCES инструкций CREATE TABLE и ALTER TABLE поддерживают предложения ON DELETE и ON UPDATE:
  - [ON DELETE { NO ACTION | CASCADE | SET NULL | SET DEFAULT }]
  - [ON UPDATE { NO ACTION | CASCADE | SET NULL | SET DEFAULT }]
- по умолчанию подразумевается действие NO ACTION (указывает, что при попытке удалить или изменить строку с ключом, на которую ссылаются внешние ключи в строках других таблиц, нужно сообщить об ошибке, а для инструкции DELETE/UPDATE выполнить откат);
- действия CASCADE, SET NULL и SET DEFAULT позволяют удалять и обновлять значения ключей, влияющие на таблицы, в которых определены связи внешних ключей, приводящие к таблице, в которую вносятся изменения:
  - CASCADE: каскадное изменение ссылающихся таблиц;
  - SET NULL: установка NULL для ссылающихся внешних ключей;
  - SET DEFAULT: установка значений по умолчанию для ссылающихся внешних ключей;
- каскадные ссылочные действия запускают триггеры AFTER UPDATE или AFTER DELETE.

Ограничения множественных каскадных действий

- последовательности каскадных ссылочных действий, запускаемые отдельными инструкциями DELETE или UPDATE, должны образовывать дерево без циклических ссылок;
- никакая таблица не должна появляться больше одного раза в списке всех каскадных ссылочных действий, вызванных инструкциями DELETE или UPDATE;
- кроме того, в дереве каскадных ссылочных действий к любой из задействованных таблиц должен быть только один путь;
- любая ветвь в дереве прерывается, как только встречается таблица, для которой указано действие NO ACTION или вообще не указано действие.

Лабораторная работа №6.
Ключи, ограничения, значения по умолчанию

1. Создать таблицу с автоинкрементным первичным ключом. Изучить функции, предназначенные для получения сгенерированного значения IDENTITY.
2. Добавить поля, для которых используются ограничения (CHECK), значения по умолчанию (DEFAULT), также использовать встроенные функции для вычисления значений.
3. Создать таблицу с первичным ключом на основе глобального уникального идентификатора.
4. Создать таблицу с первичным ключом на основе последовательности.
5. Создать две связанные таблицы, и протестировать на них различные варианты действий для ограничений ссылочной целостности (NO ACTION | CASCADE | SET NULL | SET DEFAULT).

Вопросы к экзамену

- Категории целостности данных: ссылочная целостность. Каскадные ограничения ссылочной целостности.
- Категории целостности данных: сущностная, доменная и пользовательская целостность
- Типы данных: атрибуты, категории, применение. Уникальные идентификаторы и особенности их использования.