СРЕДСТВА SQL ДЛЯ ОБЕСПЕЧЕНИЯ ОГРАНИЧЕНИЙ ЦЕЛОСТНОСТИ

САЖИН Юлиана
Coordonator: САРАНЧУК Дориан

Технический Университет Молдовы

***Аннотация**: В данной статье описываются ключевые моменты создания ограничений для таблиц базы данных, используя средства SQL. Они вводятся с целью защиты базы данных от нарушения согласованности сохраняемых в ней данных. Ограничения устанавливаются к столбцу или к таблице. Ограничение столбца применяется только к определенному полю, а ограничение таблицы - к группам из одного или более полей. К ограничениям целостности относятся ограничение первичного ключа PRIMARY KEY, ограничение внешнего ключа FOREIGN KEY, ограничение уникальности UNIQUE, ограничение значения NULL, ограничение на проверку CHECK*

***Ключевые слова:** база данных, таблица, атрибут, ограничения целостности, уникальность*

1. Введение

База данных - это структурированные знания об объектах. База данных помогает систематизировать и хранить информацию из определенной предметной области, облегчает доступ к данным, поиск и предоставление необходимых сведений. Базы данных используются в очень многих областях жизнедеятельности человека. От того насколько правильно спроектирована база данных зависит скорость и эффективность работы приложений, использующих ее. Проектирование баз данных – это сложный и длительный процесс, состоящий из нескольких этапов, требующий специальные знания у разработчика.

Непосредственное проектирование отношений базы данных происходит на этапе логического проектирования. Именно на этом этапе создается схема базы данных – схема отношений с указанием первичных ключей, а также «связей» между отношениями, представляющих собой внешние ключи, определяются типы записей. При создании отношении требуется учитывать особенности СУБД, а так же задачи, которые будут решаться на основе данных отношений.

2. Ограничения целостности

При создании баз данных большое внимание должно быть уделено средствам поддержания данных в целостном состоянии. Стандартом языка SQL предусмотрены функции, которые предназначены для поддержания целостности данных. Эта поддержка включает средства задания ограничений, они вводятся с целью защиты базы данных от нарушения согласованности сохраняемых в ней данных.

Ограничения целостности рассмотрим на небольшом примере. Создадим следующие отношения:

```
CREATE TABLE Товар (
        Код_товара      INT,
        Название        VARCHAR(50),
        Стоимость       MONEY,
        Тип             VARCHAR(50),
        Сорт            VARCHAR(30),
        Количество      SMALLINT);
CREATE TABLE Клиент(
        Фискальный_код      CHAR(13),
        Название_фирмы      VARCHAR(50),
        Адрес               VARCHAR(70),
        Телефон             CHAR(15));
CREATE TABLE Сделка(
        Фискальный_код      CHAR(13),
        Код_товара          INT,
        Дата                DATE,
        Количество          SMALLINT);
```

2.1 Ограничения первичного ключа (PRIMARY KEY)

Таблица обычно имеет столбец или комбинацию столбцов, значения которых уникально идентифицируют каждую строку в таблице. Этот столбец (или столбцы) называется первичным ключом таблицы и нужен для обеспечения ее целостности. Если в первичный ключ входит более одного столбца, то значения в пределах одного столбца могут дублироваться, но любая комбинация значений всех столбцов первичного ключа должна быть уникальна. Таблица может иметь только одно ограничение PRIMARY KEY, причем ни один из включенных в первичный ключ столбцов не может принимать значение NULL.

В современных инструментальных системах управления базами данных (СУБД) обеспечивается возможность использования до 1000 столбцов. Поэтому теоретически сложный ключ может состоять из 1000 полей, что, конечно, абсурдно. Кроме того, при выборе или назначении ключевого поля необходимо учитывать типы и размеры полей. В частности, поле типа «текст» нельзя использовать в качестве ключевого поля.

Среди атрибутов необходимо искать атрибуты с уникальными значениями или сочетания атрибутов, кортежи которых уникальны, но при этом количество атрибутов, входящих в состав составного первичного ключа, не должно превышать разумного числа, например, трех. Ввод дополнительного атрибута типа “счетчик”, во-первых, не всегда возможен, а во-вторых, не является наилучшим решением проблемы, так как в таком случае мы можем получить абсолютно одинаковые кортежи, различающиеся только значением ключа, а это ведет к избыточности информации в БД.

Лучше всего в качестве первичных ключей использовать столбцы с целочисленными значениями. Индексирование по символьным значениям (VARCHAR или CHAR) может вызвать проблем с производительностью.

Рассмотрим наш пример. В отношении Товар в качестве ключа могли бы выступать Код_товара или Название товара. Но учитывая, что в таблице могут встретиться два одноименных товара, то атрибут Название товара не может гарантировать уникальность кортежа. В то время как атрибут Код_товара будт уникальным и сможет однозначно идентифицировать запись.

В таблице Клиент первичным ключом будет Фискальный_код, так как фискальный код уникален для каждого предприятия.

А вот в таблице Сделка определить первичный ключ гораздо сложнее. Ни одно из полей не сможет обеспечить уникальность кортежа. Один и тот же клиент может много раз участвовать в сделках, так же как и один и тот же вид товара может быть продан различным клиентам. Существует два способа создания первичного ключа в таком отношении. Либо мы добавляем новое поле, предположим Номер_сделки, либо устанавливаем составной первичный ключ. В состав составного первичного ключа войдут следующие атрибуты (Фискальный_код, Код_товара, Дата). Ключ будет состоять из трех атрибутов. Один и тот же клиент может заказать один и тот же вид товара несколько раз, поэтому сочетание (Фискальный_код, Код_товара) не будет гарантировать уникальность, поэтому добавляется третий атрибут Дата. Один и тот же клиент может купить один и тот же товар несколько раз, но при этом дата оформления сделки будет различной.

Первичный ключ можно установить во время создания таблицы:

```
CREATE TABLE Клиент(
        Фискальный_код      CHAR(13) PRIMARY KEY,
        Название_фирмы      VARCHAR(50),
        Адрес               VARCHAR(70),
        Телефон             CHAR(15));
```

Либо используя команду Alter Table

```
ALTER TABLE Товар
ADD CONSTRAINT PK1 PRIMARY KEY (Код_товара);
ALTER TABLE Сделка
ADD CONSTRAINT PK3 PRIMARY KEY (Код_товара, Фискальный_код, Дата);
```

2.2 Ограничения внешнего ключа (FOREIGN KEY)

Ограничение внешнего ключа - это основной механизм для поддержания ссылочной целостности между таблицами реляционной базы данных. Столбец дочерней таблицы, определенный в качестве внешнего ключа в параметре FOREIGN KEY, применяется для ссылки на столбец родительской таблицы, являющийся в ней первичным ключом. Имя родительской таблицы и столбцы ее первичного ключа указываются в предложении REFERENCES. Данные в столбцах, определенных в

качестве внешнего ключа, могут принимать только такие же значения, какие находятся в связанных с ним столбцах первичного ключа родительской таблицы. Совпадение имен столбцов для связи дочерней и родительской таблиц необязательно. Первичный ключ может быть определен для столбца с одним именем, в то время как столбец, на который наложено ограничение FOREIGN KEY, может иметь совершенно другое имя. Единственным требованием остается соответствие столбцов по типу и размеру данных.

Внешний ключ может быть связан не только с первичным ключом другой таблицы. Он может быть определен и для столбцов с ограничением UNIQUE второй таблицы или любых других столбцов, но таблицы должны находиться в одной базе данных.

Ограничение ссылочной целостности задает требование, согласно которому для каждой записи в дочерней таблице должна иметься запись в родительской таблице. При этом изменение значения столбца связи в записи родительской таблицы при наличии дочерней записи блокируется, равно как и удаление родительской записи что гарантируется параметрами ON DELETE NO ACTION и ON UPDATE NO ACTION, принятыми по умолчанию. Для разрешения каскадного воздействия следует использовать параметры ON DELETE CASCADE и ON UPDATE CASCADE.

В нашем примере таблицы Товар и Клиент связываются через таблицу Сделка. В таблице Сделка поле Код_товара будет внешним ключом и будет обеспечивать связь с таблицей Товар. Поле Фискальный_код также будет внешним ключом и будет обеспечивать связь с таблицей Клиент. Установим в таблице Сделка внешние ключи:

```
ALTER TABLE Сделка
ADD CONSTRAINT FK1 FOREIGN KEY (Фискальный_код) REFERENCES Клиент ON DELETE
CASCADE ON UPDATE CASCADE,
CONSTRAINT FK2 FOREIGN KEY (Код_товара) REFERENCES Товар ON DELETE CASCADE
ON UPDATE CASCADE
```

Внешние ключи можно указать и во время создания таблицы.

2.3 Ограничение уникальности ключа (UNIQUE)

Это ограничение задает требование уникальности значения поля (столбца) или группы полей (столбцов), входящих в уникальный ключ, по отношению к другим записям. Ограничение UNIQUE для столбца таблицы похоже на первичный ключ: для каждой строки данных в нем должны содержаться уникальные значения. Установив для некоторого столбца ограничение первичного ключа, можно одновременно установить для другого столбца ограничение UNIQUE. Отличие в ограничении первичного и уникального ключа заключается в том, что первичный ключ служит как для упорядочения данных в таблице, так и для соединения связанных между собой таблиц. Кроме того, при использовании ограничения UNIQUE допускается существование значения NULL, но лишь единственный раз.

В нашем примере можем установить свойство уникальности для названия фирмы клиента:

```
ALTER TABLE Клиент ADD UNIQUE (Название_фирмы).
```

2.4 Ограничение на значение (NOT NULL)

Для каждого столбца таблицы можно установить ограничение NOT NULL, запрещающее ввод в этот столбец нулевого значения.

Старайтесь не использовать NULL столбцы кроме случаев, когда они вам осознанно нужны. Объявляйте везде, где возможно, столбцы как NOT NULL. Это позволяет ускорить все операции и сэкономить по одному биту для каждого столбца. Нужно просто избегать наличия NULL во всех столбцах по умолчанию. NULL не равно ни логическому значению FALSE, ни пустой строке, ни нулю. При сравнении NULL с любым значением будет получен результат NULL, а не FALSE и не 0. Более того, NULL не равно NULL!

Для нашего примера во время создания таблицы укажем, что все поля обязательны к заполнению

```
CREATE TABLE Товар (
    Код_товара    INT             NOT NULL,
    Название      VARCHAR(50) NOT NULL,
    Стоимость     MONEY           NOT NULL,
    Тип           VARCHAR(50  NOT NULL,
    Сорт          VARCHAR(30) NOT NULL,
    Количество    SMALLINT NOT NULL) ;
```

2.5 Ограничение проверочное (CHECK)

Данное ограничение используется для проверки допустимости данных, вводимых в конкретный столбец таблицы, т.е. ограничение CHECK обеспечивает еще один уровень защиты данных. Ограничения целостности CHECK задают диапазон возможных значений для столбца или столбцов. В основе ограничений целостности CHECK лежит использование логических выражений.

Допускается применение нескольких ограничений CHECK к одному и тому же столбцу. В этом случае они будут применимы в той последовательности, в которой происходило их создание. Возможно применение одного и того же ограничения к разным столбцам и использование в логических выражениях значений других столбцов.

В нашем примере можно установить правило, которое будет проверять количество оставшегося на складе товара и значение поля Сорт, которое может принимать только три значения: первый, второй и третий.

```
CREATE TABLE Товар (
    Код_товара     INT              NOT NULL,
    Название       VARCHAR(50)  NOT NULL,
    Стоимость      MONEY            NOT NULL,
    Тип      VARCHAR(50)  NOT NULL,
    Сорт     VARCHAR(30) NOT NULL CHECK (Сорт IN (‘первый’, ’второй’, ’третий’)),
    Количество SMALLINT NOT NULL CHECK( Количество>=0) );
```

2.6 Ограничение по умолчанию (DEFAULT)

Столбцу может быть присвоено значение по умолчанию. Оно будет актуальным в том случае, если пользователь не введет в столбец никакого иного значения.

Отдельно необходимо отметить пользу от использования значений по умолчанию при добавлении нового столбца в таблицу. Если для добавляемого столбца не разрешено хранение значений NULL и не определено значение по умолчанию, то операция добавления столбца закончится неудачей.

В примере можно установить значение по умолчанию для поля Адрес из таблицы Клиент.

```
CREATE TABLE Клиент(
    Фискальный_код      CHAR(13) PRIMARY KEY,
    Название_фирмы      VARCHAR(50) NOT NULL,
    Адрес               VARCHAR(70) DEFAULT (‘UNKNOWN’),
    Телефон             CHAR(15));
```

Заключение

К проектированию и созданию таблиц, а также ограничений следует отнестись очень серьезно, так как в дальнейшем это будет иметь огромное влияние на корректность и производительность работы базы данных. Лучше потратить чуть больше времени внимательно изучая все тонкости инструментов, предлагаемых СУБД, и в итоге получить систему, в которой данные будут правильными, ввод данных будет проверяться, и ресурсы системы будут использованы оптимальным образом. Выявление и исправление ошибок занимает гораздо больше времени и может привести к непредсказуемым результатам, вплоть до того, что придется заново создавать базу данных.

Библиография

1. Брешенков А. В., Белоус В. В *Метод назначения первичных ключей в информации табличного вида.* URL: http://technomag.edu.ru/doc/134299.html (Дата обращения 20.11.2013).
2. *SQL в примерах и задачах. Учеб. пособие*/И.Ф. Астахова, А.П. Толстобров, В.М. Мельников.— Мн.: Новое знание, 2002. - 176 с.
3. *SQL. Справочник, 2-е издание.*/К.Клайн, Д.Клайн, Б.Ханта — М.: КУДИЦ-ОБРАЗ, 2006 - 832с.