# Тема №8 – Ограничения.

Ограничение — это, прежде всего, формулировка требований к данным. Ограничения уста­навливаются на уровне столбца или таблицы и гарантируют соответствие данных опреде­ленным правилам обеспечения целостности данных.

**Ограничения проверки** наиболее общий тип ограничений. В его определении вы можете указать, что значение данного столбца должно удовлетворять логическому выражению (проверке истинности). Например, цену товара можно ограничить положительными значениями так:

CREATE TABLE products (

product\_no integer,

name text,

price numeric CHECK (price > 0)

);

Ограничение определяется после типа данных, как и значение по умолчанию. Значения по умолчанию и ограничения могут указываться в любом порядке. Ограничение-проверка состоит из ключевого слова CHECK, за которым идёт выражение в скобках. Это выражение должно включать столбец, для которого задаётся ограничение, иначе оно не имеет большого смысла. Вы можете также присвоить ограничению отдельное имя. Это улучшит сообщения об ошибках и позволит вам ссылаться на это ограничение, когда вам понадобится изменить его. Сделать это можно так:

CREATE TABLE products (

product\_no integer,

name text,

price numeric CONSTRAINT positive\_price CHECK (price > 0)

);

Чтобы создать именованное ограничение, напишите ключевое слово CONSTRAINT, а за ним идентификатор и собственно определение ограничения. Ограничение-проверка может также ссылаться на несколько столбцов. Например, если вы храните обычную цену и цену со скидкой, так вы можете гарантировать, что цена со скидкой будет всегда меньше обычной:

CREATE TABLE products (

product\_no integer,

name text,

price numeric CHECK (price > 0),

discounted\_price numeric CHECK (discounted\_price > 0),

CHECK (price > discounted\_price)

);

Следует заметить, что ограничение-проверка удовлетворяется, если выражение принимает значение true или NULL. Так как результатом многих выражений с операндами NULL будет значение NULL, такие ограничения не будут препятствовать записи NULL в связанные столбцы. Чтобы гарантировать, что столбец не содержит значения NULL, можно использовать ограничение NOT NULL.

**Ограничения NOT NULL** просто указывает, что столбцу нельзя присваивать значение NULL. Пример синтаксиса:

CREATE TABLE products (

product\_no integer NOT NULL,

name text NOT NULL,

price numeric

);

Если этот атрибут явным образом не будет использован, то по умолчанию столбец будет допускать значение NULL. Исключением является тот случай, когда столбец выступает в роли первичного ключа - в этом случае по умолчанию столбец имеет значение NOT NULL.

Для столбца можно определить больше одного ограничения, указав их одно за другим:

CREATE TABLE products (

product\_no integer NOT NULL,

name text NOT NULL,

price numeric NOT NULL CHECK (price > 0)

);

При проектировании баз данных чаще всего большинство столбцов должны быть помечены как NOT NULL.

**Ограничения уникальности** гарантируют, что данные в определённом столбце или группе столбцов уникальны среди всех строк таблицы.

Синтаксис в виде ограничения столбца:

CREATE TABLE products (

product\_no integer UNIQUE,

name text,

price numeric

);

Синтаксис в виде ограничения таблицы:

CREATE TABLE products (

product\_no integer,

name text,

price numeric,

UNIQUE (product\_no)

);

Чтобы определить ограничение уникальности для группы столбцов, запишите его в виде ограничения таблицы, перечислив имена столбцов через запятую:

CREATE TABLE example (

a integer,

b integer,

c integer,

UNIQUE (a, c)

);

Ограничение уникальности нарушается, если в таблице оказывается несколько строк, у которых совпадают значения всех столбцов, включённых в ограничение. Однако два значения NULL при сравнении никогда не считаются равными. Это означает, что даже при наличии ограничения уникальности в таблице можно сохранить строки с дублирующимися значениями, если они содержат NULL в одном или нескольких столбцах ограничения. Это поведение соответствует стандарту SQL, но мы слышали о СУБД, которые ведут себя по-другому.

**Ограничения Primary key**

Ограничение первичного ключа означает, что образующий его столбец или группа столбцов может быть уникальным идентификатором строк в таблице. Для этого требуется, чтобы значения были одновременно уникальными и отличными от NULL. Таким образом, таблицы со следующими двумя определениями будут принимать одинаковые данные:

CREATE TABLE products (

product\_no integer UNIQUE NOT NULL,

name text,

price numeric

);

CREATE TABLE products (

product\_no integer PRIMARY KEY,

name text,

price numeric

);

При добавлении первичного ключа автоматически создаётся уникальный индекс для столбца или группы столбцов, перечисленных в первичном ключе, и данные столбцы помечаются как NOT NULL. Таблица может иметь максимум один первичный ключ.

**Ограничения FOREING KEY**

Ограничение внешнего ключа указывает, что значения столбца (или группы столбцов) должны соответствовать значениям в некоторой строке другой таблицы. Это называется ссылочной целостностью двух связанных таблиц. Пусть у вас уже есть таблица продуктов, которую мы неоднократно использовали ранее:

CREATE TABLE products (

product\_no integer PRIMARY KEY,

name text,

price numeric

);

Предположим, что у вас есть таблица с заказами этих продуктов. Мы хотим, чтобы в таблице заказов содержались только заказы действительно существующих продуктов. Поэтому мы определим в ней ограничение внешнего ключа, ссылающееся на таблицу продуктов:

CREATE TABLE orders (

order\_id integer PRIMARY KEY,

product\_no integer **REFERENCES products (product\_no)**,

quantity integer

);

С таким ограничением создать заказ со значением product\_no, отсутствующим в таблице products (и не равным NULL), будет невозможно. В такой схеме таблицу orders называют подчинённой таблицей, а products — главной. Столбцы называют так же подчинённым и главным.

Если опустить список столбцов, внешний ключ будет неявно связан с первичным ключом главной таблицы. Вышеуказанную команду можно сократить следующим образом:

CREATE TABLE orders (

order\_id integer PRIMARY KEY,

product\_no integer **REFERENCES products**,

quantity integer

);

Внешний ключ также может ссылаться на группу столбцов. В этом случае его нужно записать в виде обычного ограничения таблицы:

CREATE TABLE t1 (

a integer PRIMARY KEY,

b integer,

c integer,

**FOREIGN KEY (b, c) REFERENCES other\_table (c1, c2)**

);

Число и типы столбцов в ограничении должны соответствовать числу и типам целевых столбцов. Таблица может содержать несколько ограничений внешнего ключа. Допустим, у вас есть таблицы продуктов и заказов, но вы хотите, чтобы один заказ мог содержать несколько продуктов (что невозможно в предыдущей схеме). Для этого вы можете использовать такую схему:

CREATE TABLE products (

product\_no integer PRIMARY KEY,

name text,

price numeric

);

CREATE TABLE orders (

order\_id integer PRIMARY KEY,

shipping\_address text,

...

);

CREATE TABLE order\_items (

product\_no integer REFERENCES products,

order\_id integer REFERENCES orders,

quantity integer,

PRIMARY KEY (product\_no, order\_id)

);

Заметьте, в последней таблице первичный ключ покрывает внешние ключи. Внешние ключи запрещают создание заказов, не относящихся ни к одному продукту. Но что делать, если после создания заказов с определённым продуктом мы захотим удалить его?

При попытке удаления продукта, на который ссылаются заказы (через таблицу order\_items), мы запрещаем эту операцию. Если же кто-то попытается удалить заказ, то удалится и его содержимое:

CREATE TABLE products (

product\_no integer PRIMARY KEY,

name text,

price numeric

);

CREATE TABLE orders (

order\_id integer PRIMARY KEY,

shipping\_address text,

...

);

CREATE TABLE order\_items (

product\_no integer REFERENCES products **ON DELETE RESTRICT**,

order\_id integer REFERENCES orders **ON DELETE CASCADE**,

quantity integer,

PRIMARY KEY (product\_no, order\_id)

);

Ограничивающие и каскадные удаления — два наиболее распространённых варианта. RESTRICT предотвращает удаление связанной строки. NO ACTION означает, что, если зависимые строки продолжают существовать при проверке ограничения, возникает ошибка (это поведение по умолчанию). (Главным отличием этих двух вариантов является то, что NO ACTION позволяет отложить проверку в процессе транзакции, а RESTRICT — нет.) CASCADE указывает, что при удалении связанных строк зависимые от них будут так же автоматически удалены. Есть ещё два варианта: SET NULL и SET DEFAULT. При удалении связанных строк они назначают зависимым столбцам в подчинённой таблице значения NULL или значения по умолчанию, соответственно. Заметьте, что это не будет основанием для нарушения ограничений. Например, если в качестве действия задано SET DEFAULT, но значение по умолчанию не удовлетворяет ограничению внешнего ключа, операция закончится ошибкой.

Аналогично указанию ON DELETE существует ON UPDATE, которое срабатывает при изменении заданного столбца. При этом возможные действия те же, а CASCADE в данном случае означает, что изменённые значения связанных столбцов будут скопированы в зависимые строки.

Обычно зависимая строка не должна удовлетворять ограничению внешнего ключа, если один из связанных столбцов содержит NULL. Если в объявление внешнего ключа добавлено MATCH FULL, строка будет удовлетворять ограничению, только если все связанные столбцы равны NULL (то есть при разных значениях (NULL и не NULL) гарантируется невыполнение ограничения MATCH FULL). Если вы хотите, чтобы зависимые строки не могли избежать и этого ограничения, объявите связанные столбцы как NOT NULL.

Внешний ключ должен ссылаться на столбцы, образующие первичный ключ или ограничение уникальности. Таким образом, для связанных столбцов всегда будет существовать индекс (определённый соответствующим первичным ключом или ограничением), а значит проверки соответствия связанной строки будут выполняться эффективно. Так как команды DELETE для строк главной таблицы или UPDATE для зависимых столбцов потребуют просканировать подчинённую таблицу и найти строки, ссылающиеся на старые значения, полезно будет иметь индекс и для подчинённых столбцов. Но это нужно не всегда, и создать соответствующий индекс можно по-разному, поэтому объявление внешнего ключа не создаёт автоматически индекс по связанным столбцам.