??????????? ????????? ??????

- Внешние ключи

??????? ?????

Для связи между таблицами применяются внешние ключи. Внешний ключ устанавливается для столбца из зависимой, подчиненной таблицы (**referencing table**), и указывает на один из столбцов из главной таблицы (**referenced table**). Как правило, внешний ключ указывает на первичный ключ из связанной главной таблицы.

????? ?????????? ?????????? ???????? ????? ?? ?????? ???????

```
REFERENCES главная_таблица (столбец_главной_таблицы)
    [ON DELETE {CASCADE|RESTRICT}]
    [ON UPDATE {CASCADE|RESTRICT}]
```

Чтобы установить связь между таблицами, после ключевого слова **REFERENCES** указывается имя связанной таблицы и далее в скобках имя столбца из этой таблицы, на который будет указывать внешний ключ. После выражения **REFERENCES** может идти выражение **ON DELETE** и **ON UPDATE**, которые уточняют поведение при удалении или обновлении данных.

????? ?????????? ?????????? ???????? ????? ?? ?????? ???????

```
FOREIGN KEY (стобец1, столбец2, ... столбецN)
    REFERENCES главная_таблица (столбец_главной_таблицы1, столбец_главной_таблицы2, ...
столбец_главной_таблицыN)
    [ON DELETE {CASCADE|RESTRICT}]
    [ON UPDATE {CASCADE|RESTRICT}]
```

Например, определим две таблицы и свяжем их посредством внешнего ключа:

```
CREATE TABLE Customers
(
    Id SERIAL PRIMARY KEY,
    Age INTEGER,
```

```
    FirstName VARCHAR(20) NOT NULL
);

CREATE TABLE Orders
(
    Id SERIAL PRIMARY KEY,
    CustomerId INTEGER REFERENCES Customers (Id),
    Quantity INTEGER
);
```

Здесь определены таблицы **Customers** и **Orders**. **Customers** является главной и представляет клиента. **Orders** является зависимой и представляет заказ, сделанный клиентом. Эта таблица через столбец **CustomerId** связана с таблицей **Customers** и ее столбцом **Id**. То есть столбец **CustomerId** является внешним ключом, который указывает на столбец **Id** из таблицы **Customers**.

Определение внешнего ключа на уровне таблицы выглядело бы следующим образом:

```
CREATE TABLE Customers
(
    Id SERIAL PRIMARY KEY,
    Age INTEGER,
    FirstName VARCHAR(20) NOT NULL
);

CREATE TABLE Orders
(
    Id SERIAL PRIMARY KEY,
    CustomerId INTEGER,
    Quantity INTEGER,
    FOREIGN KEY (CustomerId) REFERENCES Customers (Id)
);
```

ON DELETE ? ON UPDATE

С помощью выражений **ON DELETE** и **ON UPDATE** можно установить действия, которые выполняются соответственно при удалении и изменении связанной строки из главной таблицы. Для установки подобного действия можно использовать следующие опции:

- **CASCADE**: автоматически удаляет или изменяет строки из зависимой таблицы при удалении или изменении связанных строк в главной таблице.
- **RESTRICT**: предотвращает какие-либо действия в зависимой таблице при удалении или изменении связанных строк в главной таблице. То есть фактически какие-либо действия отсутствуют.
- **NO ACTION**: действие по умолчанию, предотвращает какие-либо действия в зависимой таблице при удалении или изменении связанных строк в главной таблице. И генерирует ошибку. В отличие от **RESTRICT** выполняет отложенную проверку на связанность между таблицами.
- **SET NULL**: при удалении связанной строки из главной таблицы устанавливает для столбца внешнего ключа значение **NULL**.
- **SET DEFAULT**: при удалении связанной строки из главной таблицы устанавливает для столбца внешнего ключа значение по умолчанию, которое задается с помощью атрибуты **DEFAULT**. Если для столбца не задано значение по умолчанию, то в качестве него применяется значение **NULL**.

????????? ????????

По умолчанию, если на строку из главной таблицы по внешнему ключу ссылается какая-либо строка из зависимой таблицы, то мы не сможем удалить эту строку из главной таблицы. Вначале нам необходимо будет удалить все связанные строки из зависимой таблицы. И если при удалении строки из главной таблицы необходимо, чтобы были удалены все связанные строки из зависимой таблицы, то применяется каскадное удаление, то есть опция **CASCADE** :

```
CREATE TABLE Orders
(
    Id SERIAL PRIMARY KEY,
    CustomerId INTEGER,
    Quantity INTEGER,
    FOREIGN KEY (CustomerId) REFERENCES Customers (Id) ON DELETE CASCADE
);
```

Аналогично работает выражение **ON UPDATE CASCADE**. При изменении значения первичного ключа автоматически изменится значение связанного с ним внешнего ключа. Но так как первичные ключи, как правило, изменяются очень редко, да и с принципе не рекомендуется использовать в качестве первичных ключей столбцы с изменяемыми значениями, то на практике выражение **ON UPDATE** используется редко.

????????? NULL

При установки для внешнего ключа опции **SET NULL** необходимо, чтобы столбец внешнего ключа допускал значение **NULL**:

```
CREATE TABLE Orders
(
    Id SERIAL PRIMARY KEY,
    CustomerId INTEGER,
    Quantity INTEGER,
    FOREIGN KEY (CustomerId) REFERENCES Customers (Id) ON DELETE SET NULL
);
```

?????????? ????????? ?? ??????????

```
CREATE TABLE Orders
(
    Id SERIAL PRIMARY KEY,
    CustomerId INTEGER DEFAULT 1,
    Quantity INTEGER,
    FOREIGN KEY (CustomerId) REFERENCES Customers (Id) ON DELETE SET DEFAULT
);
```

Если для столбца значение по умолчанию не задано через параметр **DEFAULT**, то в качестве такового используется значение **NULL** (если столбец допускает **NULL**).