**5.5. Изменение таблиц**

**[5.5.1. Добавление столбца](https://postgrespro.ru/docs/postgresql/9.6/ddl-alter" \l "ddl-alter-adding-a-column)**

**[5.5.2. Удаление столбца](https://postgrespro.ru/docs/postgresql/9.6/ddl-alter" \l "ddl-alter-removing-a-column)**

**[5.5.3. Добавление ограничения](https://postgrespro.ru/docs/postgresql/9.6/ddl-alter" \l "ddl-alter-adding-a-constraint)**

**[5.5.4. Удаление ограничения](https://postgrespro.ru/docs/postgresql/9.6/ddl-alter" \l "ddl-alter-removing-a-constraint)**

**[5.5.5. Изменение значения по умолчанию](https://postgrespro.ru/docs/postgresql/9.6/ddl-alter" \l "idp8)**

**[5.5.6. Изменение типа данных столбца](https://postgrespro.ru/docs/postgresql/9.6/ddl-alter" \l "idp9)**

**[5.5.7. Переименование столбца](https://postgrespro.ru/docs/postgresql/9.6/ddl-alter" \l "idp10)**

**[5.5.8. Переименование таблицы](https://postgrespro.ru/docs/postgresql/9.6/ddl-alter" \l "idp11)**

Если вы создали таблицы, а затем поняли, что допустили ошибку, или изменились требования вашего приложения, вы можете удалить её и создать заново. Но это будет неудобно, если таблица уже заполнена данными, или если на неё ссылаются другие объекты базы данных (например, по внешнему ключу). Поэтому PostgreSQL предоставляет набор команд для модификации таблиц. Заметьте, что это по сути отличается от изменения данных, содержащихся в таблице: здесь мы обсуждаем модификацию определения, или структуры, таблицы.

Вы можете:

* Добавлять столбцы
* Удалять столбцы
* Добавлять ограничения
* Удалять ограничения
* Изменять значения по умолчанию
* Изменять типы столбцов
* Переименовывать столбцы
* Переименовывать таблицы

Все эти действия выполняются с помощью команды [ALTER TABLE](https://postgrespro.ru/docs/postgresql/9.6/sql-altertable" \o "ALTER TABLE); подробнее о ней вы можете узнать в её справке.

**5.5.1. Добавление столбца**

Добавить столбец вы можете так:

ALTER TABLE products ADD COLUMN description text;

Новый столбец заполняется заданным для него значением по умолчанию (или значением NULL, если вы не добавите указание DEFAULT).

При этом вы можете сразу определить ограничения столбца, используя обычный синтаксис:

ALTER TABLE products ADD COLUMN description text CHECK (description <> '');

На самом деле здесь можно использовать все конструкции, допустимые в определении столбца в команде CREATE TABLE. Помните однако, что значение по умолчанию должно удовлетворять данным ограничениям, чтобы операция ADD выполнилась успешно. Вы также можете сначала заполнить столбец правильно, а затем добавить ограничения (см. ниже).

**Подсказка**

Добавление столбца со значением по умолчанию приводит к изменению всех строк таблицы (в них будет сохранено новое значение). Однако если значение по умолчанию не указано, PostgreSQL может обойтись без физического изменения. Поэтому, если вы планируете заполнить столбец в основном не значениями по умолчанию, лучше будет добавить столбец без значения по умолчанию, затем вставить требуемые значения с помощью UPDATE, а потом определить значение по умолчанию, как описано ниже.

**5.5.2. Удаление столбца**

Удалить столбец можно так:

ALTER TABLE products DROP COLUMN description;

Данные, которые были в этом столбце, исчезают. Вместе со столбцом удаляются и включающие его ограничения таблицы. Однако если на столбец ссылается ограничение внешнего ключа другой таблицы, PostgreSQL не удалит это ограничение неявно. Разрешить удаление всех зависящих от этого столбца объектов можно, добавив указание CASCADE:

ALTER TABLE products DROP COLUMN description CASCADE;

Общий механизм, стоящий за этим, описывается в [Разделе 5.13](https://postgrespro.ru/docs/postgresql/9.6/ddl-depend).

**5.5.3. Добавление ограничения**

Для добавления ограничения используется синтаксис ограничения таблицы. Например:

ALTER TABLE products ADD CHECK (name <> '');

ALTER TABLE products ADD CONSTRAINT some\_name UNIQUE (product\_no);

ALTER TABLE products ADD FOREIGN KEY (product\_group\_id)

REFERENCES product\_groups;

Чтобы добавить ограничение NOT NULL, которое нельзя записать в виде ограничения таблицы, используйте такой синтаксис:

ALTER TABLE products ALTER COLUMN product\_no SET NOT NULL;

Ограничение проходит проверку автоматически и будет добавлено, только если ему удовлетворяют данные таблицы.

**5.5.4. Удаление ограничения**

Для удаления ограничения вы должны знать его имя. Если вы не присваивали ему имя, это неявно сделала система, и вы должны выяснить его. Здесь может быть полезна команда psql \d ***имя\_таблицы*** (или другие программы, показывающие подробную информацию о таблицах). Зная имя, вы можете использовать команду:

ALTER TABLE products DROP CONSTRAINT some\_name;

(Если вы имеете дело с именем ограничения вида $2, не забудьте заключить его в кавычки, чтобы это был допустимый идентификатор.)

Как и при удалении столбца, если вы хотите удалить ограничение с зависимыми объектами, добавьте указание CASCADE. Примером такой зависимости может быть ограничение внешнего ключа, связанное со столбцами ограничения первичного ключа.

Так можно удалить ограничения любых типов, кроме NOT NULL. Чтобы удалить ограничение NOT NULL, используйте команду:

ALTER TABLE products ALTER COLUMN product\_no DROP NOT NULL;

(Вспомните, что у ограничений NOT NULL нет имён.)

**5.5.5. Изменение значения по умолчанию**

Назначить столбцу новое значение по умолчанию можно так:

ALTER TABLE products ALTER COLUMN price SET DEFAULT 7.77;

Заметьте, что это никак не влияет на существующие строки таблицы, а просто задаёт значение по умолчанию для последующих команд INSERT.

Чтобы удалить значение по умолчанию, выполните:

ALTER TABLE products ALTER COLUMN price DROP DEFAULT;

При этом по сути значению по умолчанию просто присваивается NULL. Как следствие, ошибки не будет, если вы попытаетесь удалить значение по умолчанию, не определённое явно, так как неявно оно существует и равно NULL.

**5.5.6. Изменение типа данных столбца**

Чтобы преобразовать столбец в другой тип данных, используйте команду:

ALTER TABLE products ALTER COLUMN price TYPE numeric(10,2);

Она будет успешна, только если все существующие значения в столбце могут быть неявно приведены к новому типу. Если требуется более сложное преобразование, вы можете добавить указание USING, определяющее, как получить новые значения из старых.

PostgreSQL попытается также преобразовать к новому типу значение столбца по умолчанию (если оно определено) и все связанные с этим столбцом ограничения. Но преобразование может оказаться неправильным, и тогда вы получите неожиданные результаты. Поэтому обычно лучше удалить все ограничения столбца, перед тем как менять его тип, а затем воссоздать модифицированные должным образом ограничения.

**5.5.7. Переименование столбца**

Чтобы переименовать столбец, выполните:

ALTER TABLE products RENAME COLUMN product\_no TO product\_number;

**5.5.8. Переименование таблицы**

Таблицу можно переименовать так:

ALTER TABLE products RENAME TO items;