ГЛЕБ ЗАЙЦЕВ

СТРОКИ В SQL

18 задач для чистки данных

```
SELECT lower(trim(email)) AS email,
       replace(phone, ' ', '') AS phone
FROM clients
WHERE email LIKE '%@%';
```

trim · lower · replace · substr

email · телефоны · артикулы · теги

все запросы запускаются в SQLite

Глеб Зайцев

Строки в SQL. 18 задач для чистки данных

«Автор»
2026

Зайцев Г.

Строки в SQL. 18 задач для чистки данных / Г. Зайцев — «Автор», 2026

Практическая книга по строковым операциям SQL для аналитиков и начинающих разработчиков. Внутри 18 задач на маленьких таблицах: trim, lower, upper, replace, substr, instr, length, LIKE, нормализация email, телефонов, артикулов, промокодов, тегов и финальный отчёт качества строк. Все примеры запускаются в SQLite, данные встроены прямо в книгу.

Содержание

Глеб Зайцев
Строки в SQL. 18 задач для чистки данных

Перед первой задачей

Почему строки портят отчёты

В реальных данных ошибка часто выглядит как лишний пробел, разный регистр, дефис в телефоне или артикул с неожиданным префиксом.

Такие мелочи ломают группировку и соединения: один и тот же email превращается в две строки, а одинаковые товары не склеиваются по ключу.

Эта книга тренирует прикладную чистку строк: привести, найти, отрезать, заменить и проверить результат на маленьких понятных таблицах.

Что важно помнить

Строковые функции зависят от СУБД сильнее, чем базовый SELECT. Примеры здесь запускаются в SQLite, а при переносе в другую систему нужно сверить названия функций.

Нормализация не должна прятать плохие данные. Если строка стала красивой после replace, всё равно полезно знать, сколько таких исправлений было.

Перед автоматической чисткой сначала сделайте маленький отчёт: исходное значение, нормализованное значение и причина изменения.

Маршрут по задачам

Практические задачи

Задача 1. Убрать пробелы вокруг email

Рабочий вопрос

Нужно привести email к аккуратному виду, если при вводе пользователь оставил пробелы в начале или конце строки.

Данные

```
CREATE TABLE clients(id INTEGER, email TEXT);
INSERT INTO clients VALUES
(1,' anna@example.test '),
(2,'oleg@example.test'),
(3,' nina@example.test');
```

Запрос

```
SELECT id, trim(email) AS clean_email
FROM clients
ORDER BY id;
```

Ожидаемый результат

```
[[1, "anna@example.test"], [2, "oleg@example.test"], [3, "nina@example.test"]]
```

Почему работает

trim(email) убирает пробелы по краям строки.
Внутренняя часть email не меняется.
ORDER BY сохраняет понятный порядок клиентов.

Где легко ошибиться

trim не удаляет пробелы внутри строки. Если пробел стоит в середине email, нужна отдельная проверка качества.

Самостоятельная проверка

Почему клиент 2 не изменился после trim, хотя функция применена ко всем строкам?

Ответ: У клиента 2 уже не было пробелов по краям email, поэтому результат совпал с исходной строкой.

Задача 2. Привести email к нижнему регистру

Рабочий вопрос

Нужно убрать различия регистра, чтобы одинаковые email не считались разными клиентами.

Данные

```
CREATE TABLE clients(id INTEGER, email TEXT);
INSERT INTO clients VALUES
(1,'Anna@Example.Test'),
(2,'oleg@example.test'),
(3,'NINA@EXAMPLE.TEST');
```

Запрос

```
SELECT id, lower(email) AS normalized_email
FROM clients
ORDER BY id;
```

Ожидаемый результат

```
[[1, "anna@example.test"], [2, "oleg@example.test"], [3, "nina@example.test"]]
```

Почему работает

lower приводит буквы к нижнему регистру.
Уже нижний email остаётся без изменений.
Нормализованный email удобнее использовать как ключ.

Где легко ошибиться

Нормализация регистра полезна для email, но не каждое текстовое поле можно бездумно переводить в lower.

Самостоятельная проверка

Почему lower помогает перед поиском дублей по email в клиентской базе?
Ответ: Так записи с разным регистром букв сравниваются как одно и то же значение.

Задача 3. Найти email с символом @

Рабочий вопрос

Нужно быстро отделить строки, которые хотя бы похожи на email, от явно неправильных значений.

Данные

```
CREATE TABLE contacts(id INTEGER, email TEXT);
INSERT INTO contacts VALUES
(1,'a@example.test'),
(2,'wrong-email'),
(3,'n@example.test'),
(4,'empty');
```

Запрос

```
SELECT id, email
FROM contacts
```

Конец ознакомительного фрагмента.