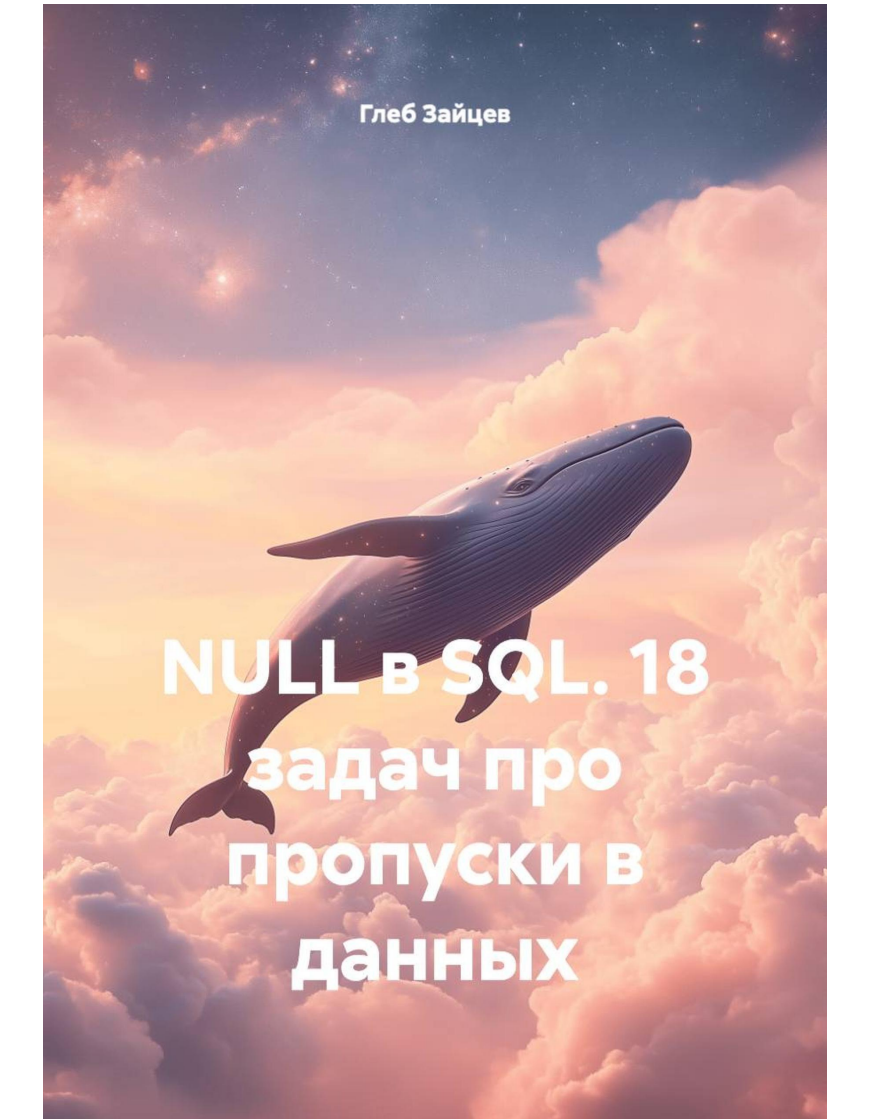


A large blue whale is depicted swimming upwards through a sky filled with soft, pinkish-orange clouds. The sky transitions from a deep blue at the top to a warm orange near the horizon. Numerous small, bright stars are scattered across the sky, some appearing to be part of the whale's body, suggesting a celestial or dreamlike theme.

Глеб Зайцев

NULL в SQL. 18 задач про пропуски в данных

Глеб Зайцев
NULL в SQL. 18 задач
про пропуски в данных

<https://litres.ru/74419432>

SelfPub; 2026

Аннотация

Практическая книга по NULL в SQL для тех, кто строит отчёты и не хочет терять строки из-за пропусков. Внутри 18 задач на маленьких таблицах: IS NULL, IS NOT NULL, COUNT, COALESCE, AVG, CASE WHEN, LEFT JOIN, NOT IN, сортировка пустых дат, группировка неизвестных значений, сравнение NULL и финальный отчёт качества данных. Все примеры запускаются в SQLite, данные встроены прямо в книгу.

Содержание

Перед первой задачей	4
Почему NULL не равен пустому месту	4
Как проверять запросы с NULL	5
Маршрут по задачам	6
Практические задачи	7
Задача 1. Найти клиентов без телефона	7
Задача 2. Оставить только заполненные email	10
Задача 3. Сравнить COUNT(*) и COUNT(column)	12
Конец ознакомительного фрагмента.	14

Глеб Зайцев

NULL в SQL. 18 задач

про пропуски в данных

Перед первой задачей

Почему NULL не равен пустому месту

NULL в SQL означает неизвестное или отсутствующее значение, а не ноль, не пустую строку и не обычный текст.

Из-за NULL отчёты часто выглядят правдоподобно, но считают не те строки: COUNT меняется, AVG пропускает значения, фильтр не находит неизвестные даты.

Эта книга тренирует не синтаксис ради синтаксиса, а рабочие ситуации: контакты клиентов, пропущенные суммы, отсутствующие связи и контроль качества данных.

Как проверять запросы с NULL

Перед запуском запроса отдельно выпишите строки с NULL. Если вы не знаете, что с ними произойдёт, отчёт ещё рано отдавать дальше.

Сравнения через знак равенства не работают с NULL так же, как с обычными значениями. Для проверки отсутствия используйте IS NULL и IS NOT NULL.

COALESCE полезен для вывода и расчётов, но он не должен прятать проблему качества данных там, где нужно честно показать пропуск.

Маршрут по задачам

1. Найти клиентов без телефона
2. Оставить только заполненные email
3. Сравнить COUNT(*) и COUNT(column)
4. Показать замену для пустого отдела
5. Посчитать сумму с пропущенной скидкой
6. Увидеть, как AVG пропускает NULL
7. Отметить строки с пропущенной суммой
8. Найти отсутствующую связь через LEFT JOIN
9. Обойти ловушку NOT IN с NULL
10. Отсортировать пустые даты в конец
11. Сгруппировать неизвестные категории
12. Посчитать разные значения вместе с NULL
13. Проверить сравнение двух NULL
14. Разделить ноль и NULL в остатках
15. Посчитать заполненность полей
16. Собрать безопасную строку адреса
17. Найти заказы без даты оплаты
18. Собрать финальный отчёт качества данных

Практические задачи

Задача 1. Найти клиентов без телефона

Рабочий вопрос

Нужно вывести клиентов, у которых телефон отсутствует, чтобы менеджер мог запросить недостающий контакт.

Данные

```
CREATE TABLE clients(id INTEGER, name  
TEXT, phone TEXT);  
INSERT INTO clients VALUES  
(1, 'Анна', '+79990000001'),  
(2, 'Илья', NULL),  
(3, 'Нина', '+79990000003'),  
(4, 'Олег', NULL);
```

Запрос

```
SELECT id, name  
FROM clients  
WHERE phone IS NULL  
ORDER BY id;
```

Ожидаемый результат

```
[[2, "Илья"], [4, "Олег"]]
```

Почему работает

IS NULL проверяет именно отсутствующее значение.

У клиентов 2 и 4 телефон равен NULL.

ORDER BY делает порядок результата предсказуемым.

Где легко ошибиться

Условие phone = NULL не найдёт такие строки, потому что NULL не сравнивается через обычное равенство.

Самостоятельная проверка

Почему в этом запросе нельзя заменить IS NULL на обычное сравнение через знак равенства?

Ответ: Потому что NULL означает неизвестное значение, и сравнение phone = NULL не становится истинным.

Задача 2. Оставить только заполненные email

Рабочий вопрос

Нужно получить список клиентов, у которых email действительно указан хотя бы как непустое значение.

Данные

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

Запрос

```
SELECT id, email  
FROM clients  
WHERE email IS NOT NULL
```

```
AND email <> ''  
ORDER BY id;
```

Ожидаемый результат

```
[[1, "a@example.test"], [4,  
"n@example.test"]]
```

Почему работает

IS NOT NULL убирает отсутствующий email.
email <> "" дополнительно убирает пустую строку.
В результате остаются только строки 1 и 4.

Где легко ошибиться

Пустая строка и NULL — разные значения.
COUNT(email) посчитает пустую строку как заполненное поле.

Самостоятельная проверка

Какая часть условия убирает пустую строку, если она формально не является NULL?

Ответ: Пустую строку убирает проверка email <> "".

Задача 3. Сравнить COUNT(*) и COUNT(column)

Рабочий вопрос

Нужно понять, сколько всего строк в таблице и сколько строк имеют заполненную сумму заказа.

Данные

```
CREATE TABLE orders(id INTEGER, total  
INTEGER);  
INSERT INTO orders VALUES  
(1,500),  
(2,NULL),  
(3,300),  
(4,NULL);
```

Запрос

```
SELECT COUNT(*) AS rows_count,  
COUNT(total) AS known_total_count  
FROM orders;
```

Ожидаемый результат

`[[4, 2]]`

Почему работает

`COUNT(*)` считает все четыре строки.

Конец ознакомительного фрагмента.

Текст предоставлен ООО «Литрес».

Прочитайте эту книгу целиком, [купив полную легальную версию](#) на Литрес.

Безопасно оплатить книгу можно банковской картой Visa, MasterCard, Maestro, со счета мобильного телефона, с платежного терминала, в салоне МТС или Связной, через PayPal, WebMoney, Яндекс.Деньги, QIWI Кошелек, бонусными картами или другим удобным Вам способом.