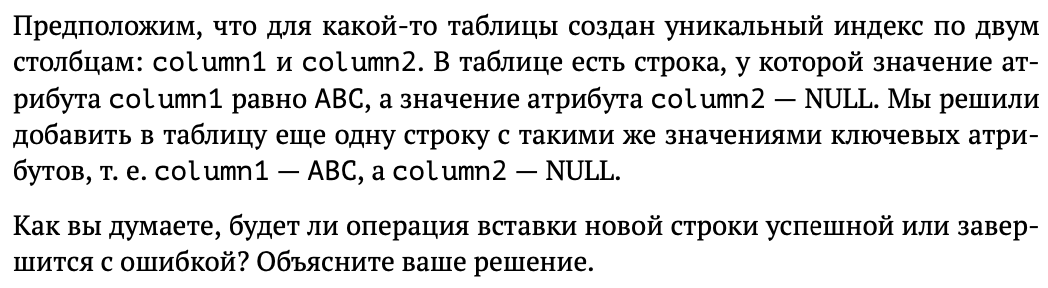
Домашняя работа №7

Попов Матвей, М8О-114СВ-24

# Задание 1



## Ответ

Операция завершится успешно, так как в PostgreSQL все значения NULL рассматриваются как разные, и уникальный индекс не будет рассматривать NULL в новой строке и в уже существуюшей как одинаковое значение. Поэтому вставка пройдет успешно, хоть значения в column1 и одинаковые, PostgreSQL воспримет NULL в column2 как разные значения.

# Задание 3

Известно, что индекс значительно ускоряет работу, если при выполнении за-

проса из таблицы отбирается лишь небольшая часть строк. Если же эта доля

велика, скажем, половина строк или более, то большого положительного эффек-

та от наличия индекса уже не будет, а возможно даже, что не будет практически

никакого эффекта. Наша задача — проверить это утверждение на практике.

Обратимся к таблице «Перелеты» (ticket\_flights). В ней имеется столбец

«Класс обслуживания» (fare\_conditions), который отличается от остальных

тем, что в нем могут присутствовать лишь три различных значения: Comfort,

Business и Economy.

Если секундомер в утилите psql выключен, то включите его.

Выполните запросы, подсчитывающие количество строк, в которых атрибут

fare\_conditions принимает одно из трех возможных значений. Каждый из

запросов выполните три-четыре раза, поскольку время может немного изме-

няться, и подсчитайте среднее время. Обратите внимание на число строк, ко-

торые возвращает функция count для каждого значения атрибута. При этом

среднее время выполнения запросов для трех различных значений атрибута

fare\_conditions будет различаться незначительно, поскольку в каждом слу-

чае СУБД просматривает все строки таблицы.

**SELECT count( \* )**

**FROM ticket\_flights**

**WHERE fare\_conditions ='Comfort';**

**SELECT count( \* )**

**FROM ticket\_flights**

**WHERE fare\_conditions ='Business';**

**SELECT count( \* )**

**FROM ticket\_flights**

**WHERE fare\_conditions ='Economy';**

Создайте индекс по столбцу fare\_conditions. Конечно, в реальной ситуации

такой индекс вряд ли целесообразен, но нам он нужен для экспериментов.

Проделайте те же эксперименты с таблицей ticket\_flights. Будет ли разли-

чаться среднее время выполнения запросов для различных значений атрибута

fare\_conditions? Почему это имеет место?

В завершение этого упражнения отметим, что в случае ошибки планировщи-

ка при использовании индекса возможно не только отсутствие положительного

эффекта, но и значительный отрицательный эффект.

## Запрос

SELECT count( \* )

FROM ticket\_flights

WHERE fare\_conditions ='Comfort';

SELECT count( \* )

FROM ticket\_flights

WHERE fare\_conditions ='Business';

SELECT count( \* )

FROM ticket\_flights

WHERE fare\_conditions ='Economy';

CREATE INDEX fare\_conditions\_index

ON ticket\_flights (fare\_conditions);

SELECT count( \* )

FROM ticket\_flights

WHERE fare\_conditions ='Comfort';

SELECT count( \* )

FROM ticket\_flights

WHERE fare\_conditions ='Business';

SELECT count( \* )

FROM ticket\_flights

WHERE fare\_conditions ='Economy';

## Результат

До добавления индекса:

- Comfort – 136ms

- Business – 75ms

- Economy – 76ms

После добавления индекса:

- Comfort – 7ms

- Business – 14ms

- Economy – 49ms

Таким образом, чем большая часть записей попадает в выборку, тем медленнее происходит операция с помощью индекса. В случае с Economy выборка составила примерно 90% всей таблицы, и с использованием индекса именно она выполнялась дольше всего. Выборка по Comfort и Business составляла в несколько раз меньше, и выполнялась на порядок быстрее. В то же время выборка без индекса заняла примерно одинаковое время, потому что в таком случае будут проверены все строки в таблице, независимо от того, какие мы ищем.