

Оптимизация SQL запросов

Explain

Как нам проверить эффективность индекса, и вообще, эффективность запроса? Может не нужно делать индекс и всё в порядке?

Можно ли ускорить запрос с помощью индекса? Как определить, страдает ли производительность запросов? Прежде чем перейти к созданию индексов или оптимизации запросов, важно понимать, что решение должно быть направлено на конкретную проблему производительности, а не быть сделано "наугад" или "потому что так надо".

Проблемы начинаются тогда, когда есть высокая нагрузка или очень большие объёмы данных. Тогда запросы выполняются очень медленно, особенно если сразу много пользователей пытаются выполнить запросы.

Как понять что есть проблема? Проанализируйте время выполнения запросов, если оно больше 200мс для обычных SQL запросов и более 10 секунд для аналитических запросов, возможно вам нужно заняться оптимизацией. Почему возможно? Потому что все цифры очень индивидуальны и зависят именно от вашего контекста, инфраструктуры, логики приложения, структур данных и так далее.

Когда возникает подозрение на проблемы с производительностью в базе данных, важно подходить к этому вопросу систематически. Один из эффективных способов – использование анализатора запросов PostgreSQL для диагностики и решения проблемы. Вот как это можно сделать, шаг за шагом.

1. Идентификация медленных запросов

Начните с определения, какие именно запросы работают медленно. Это можно сделать, включив логирование медленных запросов в PostgreSQL. Например, если вы настроите логирование всех запросов, которые выполняются дольше 200 миллисекунд, вы сможете увидеть, какие запросы занимают больше всего времени.

2. Использование EXPLAIN

Для каждого медленного запроса используйте команду `EXPLAIN` или `EXPLAIN ANALYZE` . Это позволит вам увидеть план выполнения запроса, который PostgreSQL использует для получения данных. Например, если вы запустите `EXPLAIN SELECT * FROM orders WHERE customer_id = 123;` , вы увидите, использует ли запрос индексы, сколько строк он сканирует, и какие операции выполняются.

3. Анализ плана запроса

Внимательно изучите полученный план. Основные моменты, на которые стоит обратить внимание:

1. Сканирование таблиц: Использует ли запрос полное сканирование таблицы? Это может быть признаком того, что индекс не используется или отсутствует.
2. Операции соединения: Какие типы соединений используются, и насколько эффективны они?
3. Оценка стоимости: PostgreSQL предоставляет оценку "стоимости" запроса, которая может помочь определить, где именно возникают задержки.

4. Оптимизация запроса

На основе анализа плана выполнения, вы можете принять меры для оптимизации запроса. Это может включать в себя добавление индексов, изменение способа соединения таблиц или даже переписывание запроса для более эффективной работы.

5. Проверка что показатели улучшились

Убедитесь, что оптимизация сработала и всё в порядке. И что, самое главное, вы не ухудшили что-то другое, исправив текущую проблему.

Пример:

Представим, что после включения логирования медленных запросов вы обнаружили, что запрос `SELECT * FROM employees WHERE department = 'HR';` занимает значительно больше времени, чем ожидалось. Используя `EXPLAIN`, вы выясняете, что происходит полное сканирование таблицы `employees`. Добавление индекса к столбцу `department` может значительно улучшить производительность. После добавления индекса повторный запуск `EXPLAIN` показывает, что запрос теперь использует этот индекс, что снижает время выполнения.

***мы дали базовое понимание и тактику для оптимизации запросов SQL к вашей базе данных. Но мы не будем учить вас SQL в данном курсе, уже есть достаточно материалов в открытом доступе (курсы\книги) для изучения этой темы, ведь это просто язык программирования, азы которого вам нужно (если нужно) изучить и отработать на практике.