УДК 004.657

ПОВЫШЕНИЕ ЭФФЕКТИВНОСТИ ЗАПРОСОВ К POSTGRESQL С ИСПОЛЬЗОВАНИЕМ ПРОМЕЖУТОЧНЫХ ТАБЛИЦ В ДЕНОРМАЛИЗОВАННОЙ ФОРМЕ

Лисицина В.В. (Университет ИТМО)
Научный руководитель – преподаватель факультета программной инженерии и компьютерной техники, Николаев В.В.
(Университет ИТМО)

Введение. При нормализации базы данных происходит создание новых таблиц, что приводит к снижению эффективности выборки данных в связи с увеличением количества соединений. В работе изучаются способы повышения производительности запросов к базе данных посредством создания промежуточного представления, соответствующего части изначальной модели в денормализованной форме в PostgreSQL.

Основная часть. Для устранения избыточности и противоречивости данных, а также обеспечения их целостности используется нормализация, которая представляет собой метод разделения данных на многочисленные таблицы по определенным правилам. Из-за этого запросы к нормализованной базе данных содержат большое число соединений, что приводит к временным затратам при их выполнении, так как соединение - достаточно ресурсоемкая операция. Процессом, обратным нормализации, является денормализация. Денормализация заключается в объединении нескольких таблиц в одну, что повышает эффективность обращения к данным. Но в то же время, денормализация может привести к следующим проблемам:

1. Риск потери целостности данных, так как данные хранятся в нескольких таблицах и изменения в одной таблице могут повлечь изменения в других таблицах. Это может привести к ошибкам и некорректным результатам при выполнении запросов.
2. Возможность возникновения ряда аномалий, таких как — аномалии обновления, вставки и удаления. Они приводят к некорректным результатам при выполнении запросов и потере целостности данных.
3. Вероятность появления повторяющихся данных, что может привести к увеличению размера базы данных и затруднить ее управление.

В рамках работы были произведены эксперименты, в ходе которых производились измерения времени выполнения запросов для нескольких таблиц, соединенных внешними ключами, и аналогичной денормализованной таблицей. В ходе тестирования выяснилось, что в большинстве случаев время выполнения запросов к денормализованной таблице в несколько раз меньше, чем для аналогичного запроса с соединениями, применяемого к нормализованной таблице. Это указывает на эффективность объединения нескольких таблиц для экономии времени, затраченного на извлечение данных.

В данной работе будет разработано решение, которое позволит увеличить производительность выполнения запросов в PostgreSQL при использовании денормализованных данных, одновременно сохраняя целостность базы данных и предотвращая возникновение аномалий. Для этого будет использоваться промежуточное представление в денормализованном виде, к которому будут перенаправляться часто выполняемые запросы, что позволит сократить время выполнения запросов.

В рамках реализации предлагается два модуля - PgGateway и PgConfig. Модуль PgConfig предназначен для генерации скриптов, которые создают денормализованное представление указанных таблиц, а также триггеров, отвечающих за обновление и создание

данных в измененных таблицах. Для создания денормализованных таблиц из нормализованных, модуль включает в себя функциональность для определения структуры таблиц, создания новых таблиц, заполнения их данными и установки правил обновления. Модуль PgGateway - Оотвечает за реконструкцию запросов, относящихся к нескольким таблицами, то есть содержащих join, в аналогичные запросы к вспомогательным денормализованным таблицам. Модуль работает следующим образом: он принимает запросы на вход, затем производит реконструкцию запросов, осуществляя замену соединений оригинальных таблиц, которые используются в изначальном запросе, на обращения к вспомогательным денормализованным таблицам. При этом, в зависимости от построенных денормализованных представлений соединения из изначального запроса могут быть убраны или заменены на соединения денормализованных таблиц. После этого полученный запрос отправляется к целевой БД.

Вывод. Представленное решение позволит более эффективно работать с запросами на чтение данных, содержащими большое количество соединений и при этом не нарушит изначальную модель данных, доступную пользователям.

Список использованных источников:

1. Шениг Г. Ю. PostgreSQL 11 Мастерство разработки // Г. Ю. Шенинг — Москва, 2019. 352 с.
2. Кириллов В.В. Введение в реляционные базы данных // В. В.Кириллов, Г.Ю Громов — Санкт-Петербург, 2009. 450 с.
3. Denormalization strategies for data retrieval from data warehouses [Электронный ресурс] Режим доступа: https://www.sciencedirect.com/science/article/pii/S0167923604003021, свободный (Дата обращения: 10.02.2022).