

Гасанов Г.М.,

Студент магистратуры

2 курс, факультет «Компьютерных Технологий и Энергетики»

Дагестанский Государственный Технический Университет

Россия, г. Махачкала

ОПТИМИЗАЦИЯ ГЕОПРОСТРАНСТВЕННЫХ ЗАПРОСОВ В ВЫСОКОНАГРУЖЕННЫХ LBS-СЕРВИСАХ НА БАЗЕ СУБД POSTGRESQL И РАСШИРЕНИЯ POSTGIS

Аннотация: *Статья посвящена проблеме обеспечения высокой производительности и масштабируемости в современных геолокационных сервисах (LBS). В работе исследуются алгоритмические и архитектурные подходы к оптимизации пространственных запросов с использованием СУБД PostgreSQL и расширения PostGIS. Подробно рассматривается механизм работы пространственных индексов на базе обобщенных деревьев поиска (GiST) и их преимущество перед полным сканированием таблиц при вычислении дистанций. На примере высоконагруженного рекомендательного сервиса распределения ресурсов (фудшеринга) представлен сравнительный анализ производительности SQL-запросов (EXPLAIN ANALYZE) и результаты нагрузочного тестирования, подтверждающие кратное снижение задержек при обработке конкурентных геозапросов.*

Ключевые слова: *LBS-сервисы, PostgreSQL, PostGIS, GiST-индекс, пространственные базы данных, оптимизация запросов, R-деревья.*

Annotation: *The article is devoted to the problem of ensuring high performance and scalability in modern location-based services (LBS). The paper explores algorithmic and architectural approaches to optimizing spatial queries using the PostgreSQL DBMS and the PostGIS extension. The mechanism of spatial indexes based on Generalized Search Trees (GiST) and their advantage over full*

table scans in distance calculation are considered in detail. Using the example of a high-load resource distribution recommendation service (foodsharing), a comparative performance analysis of SQL queries (EXPLAIN ANALYZE) and load testing results are presented, confirming a multiple reduction in latency when processing concurrent spatial queries.

Key words: *LBS services, PostgreSQL, PostGIS, GiST index, spatial databases, query optimization, R-trees.*

Введение

Современный этап развития информационных технологий характеризуется взрывным ростом популярности геолокационных сервисов (Location-Based Services, LBS). К данному классу относятся агрегаторы такси, сервисы доставки, приложения для каршеринга и развивающиеся платформы экономики совместного потребления, такие как системы фудшеринга. Ключевой вычислительной задачей в архитектуре подобных приложений является непрерывный поиск ближайших доступных объектов (ближайшего свободного автомобиля, ресторана или пункта выдачи излишков продуктов) в условиях постоянного обновления координат как самих объектов, так и пользователей-реципиентов.

При проектировании архитектуры LBS-сервисов разработчики сталкиваются с фундаментальной проблемой масштабируемости базы данных (БД). Стандартные реляционные подходы и индексы (например, B-Tree) не приспособлены для работы с многомерными пространственными данными. Вычисление расстояний между координатами на сфере (с использованием формулы гаверсинуса) для каждой записи в базе данных приводит к полному сканированию таблиц (Full Table Scan), что при росте числа записей до сотен тысяч вызывает критическую деградацию времени отклика сервиса. Актуальность данного исследования обусловлена необходимостью поиска и внедрения эффективных методов индексирования и маршрутизации

геозапросов для обеспечения работы сервисов в режиме реального времени (Near Real-Time).

Целью данной работы является исследование алгоритмов пространственной оптимизации в СУБД PostgreSQL с использованием модуля PostGIS и оценка их эффективности в условиях высоких конкурентных нагрузок.

Теоретические основы работы с геопространственными данными

Для решения задачи пространственного поиска объектов в заданном радиусе R от пользователя с координатами (lat, lon) наивный алгоритм предполагает вычисление математической функции расстояния “d” для всех N объектов в системе. Вычислительная сложность такого подхода составляет $O(N)$, что делает систему неработоспособной при росте базы данных.

Решение проблемы достигается за счет перехода от точных вычислений ко всем точкам к двухэтапной фильтрации с использованием структуры ограничивающих прямоугольников (Bounding Box). В основе пространственной индексации в PostGIS лежат R-деревья (Region Trees), реализованные через механизм GiST (Generalized Search Tree).

R-дерево разбивает географическое пространство на иерархически вложенные, пересекающиеся прямоугольные области. Корневой узел дерева содержит прямоугольники, охватывающие более мелкие области, которые, в свою очередь, содержат конкретные географические точки (листья дерева). При выполнении запроса СУБД сначала проверяет пересечение области поиска с крупными ограничивающими прямоугольниками (Index Scan). Если пересечения нет, целые ветви дерева мгновенно отсекаются от поиска, что снижает алгоритмическую сложность выборки кандидатов до $O(\log N)$. Только для объектов, прошедших первичную фильтрацию по прямоугольникам, применяется точный (и ресурсоемкий) расчет расстояния на сфере.

Архитектурная и программная реализация в СУБД PostgreSQL

В рамках практической реализации базы данных для системы распределения продуктов (фудшеринга) был использован тип данных geography (Point, 4326), предоставляемый расширением PostGIS. В отличие от типа geometry, который оперирует абстрактными декартовыми координатами на плоскости, тип geography учитывает кривизну Земли (эллипсоид WGS 84), что позволяет выполнять запросы с указанием радиуса непосредственно в метрах, избегая сложных математических трансформаций на уровне приложения.

Для поля location в таблице заявок на продукты food_items был создан специализированный пространственный индекс:

```
CREATE INDEX food_items_location_gist_idx  
ON food_items USING GIST (location);
```

Синтаксис SQL-запросов также играет критическую роль в активации механизма индексов. Использование классических математических функций внутри блока WHERE блокирует использование GiST-индекса. Рассмотрим два варианта реализации запроса поиска продуктов в радиусе 5000 метров:

Вариант 1 (Неоптимизированный, вызывает Seq Scan):

```
SELECT id, title FROM food_items  
WHERE ST_DistanceSphere(location, ST_MakePoint(47.5,  
42.9)::geography) < 5000;
```

Данный запрос заставляет СУБД извлечь каждую строку и вычислить функцию ST_DistanceSphere до фильтрации.

Вариант 2 (Оптимизированный, вызывает Bitmap Index Scan):

```
SELECT id, title FROM food_items  
WHERE ST_DWithin(location, ST_MakePoint(47.5, 42.9)::geography,  
5000);
```

Функция ST_DWithin разработана специально для работы с GiST-индексами. На первом этапе она автоматически строит Bounding Box вокруг точки пользователя размером 5x5 км, извлекает кандидатов по индексу, и

лишь затем вычисляет точное расстояние для отсечения объектов в «углах» прямоугольника.

Вычислительный эксперимент и нагрузочное тестирование

Для доказательства эффективности предложенной архитектуры базы данных был проведен вычислительный эксперимент. Тестовый стенд включал СУБД PostgreSQL 15 и серверное приложение на базе асинхронного фреймворка FastAPI. База данных была заполнена синтетическим набором из 100 000 записей (карточек продуктов) со случайным распределением координат в пределах крупного региона.

Первично был проведен анализ плана выполнения запросов с помощью команды EXPLAIN ANALYZE. При использовании неоптимизированного подхода (Вариант 1) время выполнения (Execution Time) одного запроса составило в среднем 145 мс. После применения функции ST_DWithin и GiST-индекса время выполнения аналогичного запроса сократилось до 4–6 мс.

На втором этапе проводилось нагрузочное тестирование с помощью фреймворка Locust для оценки поведения системы в условиях реальной эксплуатации (ситуация, когда сотни пользователей одновременно открывают мобильное приложение и запрашивают ленту ближайших продуктов). Постепенно наращивалась нагрузка от 10 до 1000 конкурентных пользователей (Virtual Users).

Результаты тестирования показали:

1. При использовании полного сканирования таблицы система начала деградировать уже при 150 одновременных пользователях (RPS), время отклика 95-го перцентиля (95th percentile latency) превысило 1200 мс, а загрузка процессора сервера базы данных достигла 100%.

2. При активации пространственной оптимизации с PostGIS система стабильно обрабатывала нагрузку в 800–1000 конкурентных запросов. 95-й перцентиль задержки зафиксировался на уровне 85 мс, что обеспечивает мгновенную загрузку интерфейса клиентского приложения. Загрузка CPU

сервера БД не превышала 35%, так как основная работа по фильтрации выполнялась на уровне бинарного дерева индекса в оперативной памяти.

Заключение

Разработка масштабируемых геолокационных сервисов невозможна без глубокого понимания механизмов хранения и индексирования пространственных данных. Проведенное исследование подтверждает, что использование стандартных подходов реляционных баз данных для расчета геопространственных дистанций приводит к эффекту бутылочного горлышка (bottleneck) при росте объема данных и пользовательской нагрузки. Интеграция расширения PostGIS в экосистему PostgreSQL и применение индексов класса GiST в сочетании с оптимизированными SQL-функциями (такими как ST_DWithin и операторы поиска k-ближайших соседей) позволяют снизить вычислительную сложность геозапросов до логарифмической.

Результаты нагрузочного тестирования наглядно демонстрируют, что предложенная архитектура обеспечивает кратное снижение времени отклика и стабильную работу LBS-сервиса даже в условиях высоких конкурентных нагрузок. Данные архитектурные решения образуют надежный технологический фундамент для реализации сложных контекстно-зависимых систем, таких как платформы фудшеринга, где скорость формирования локальной выборки критически важна для предотвращения потерь ресурсов с истекающим сроком годности.

Использованные источники:

1. Оби, С. Разработка геоинформационных систем на базе PostgreSQL и PostGIS / С. Оби. — Москва : ИНФРА-М, 2024. — 198 с.
2. Маркова, А. В. Индексирование многомерных данных в реляционных СУБД: применение R-деревьев / А. В. Маркова // Вестник информационных технологий. — 2023. — № 5. — С. 78–85.

3. Таненбаум, Э. Распределенные системы. Принципы и парадигмы / Э. Таненбаум, М. ван Стеен. — Санкт-Петербург : Питер, 2021. — 736 с.
4. Смит, Д. Высокопроизводительный Python: приложения на FastAPI и асинхронное программирование / Д. Смит ; пер. с англ. — Санкт-Петербург : Питер, 2023. — 384 с.
5. PostGIS Reference Documentation [Электронный ресурс]. — URL: <https://postgis.net/documentation/> (дата обращения: 10.05.2026).
6. PostgreSQL: Documentation: 15: GiST Indexes [Электронный ресурс]. — URL: <https://www.postgresql.org/docs/15/gist.html> (дата обращения: 10.05.2026).