УДК 004.657

СРАВНИТЕЛЬНЫЙ АНАЛИЗ РЕШЕНИЙ ХРАНЕНИЯ ДАННЫХ ДЛЯ СИСТЕМЫ СОВМЕСТНОГО РЕДАКТИРОВАНИЯ В РЕАЛЬНОМ ВРЕМЕНИ

Бессонов В.Ю. (ИТМО)
Научный руководитель – кандидат технических наук Жданов А.Д.
(ИТМО)

Введение. Интерактивные веб-сервисы совместного редактирования, примером которых является проект Reddit r/place, предъявляют специфические требования к подсистеме хранения данных: высокая пропускная способность записи при аппендитивном характере нагрузки, низкая задержка чтения последнего состояния, а также возможность чтения большой выборки при инициализации клиента. В рамках проекта PixelBattle — реализации интерактивного пиксельного полотна для студентов Университета ИТМО — был проведён сравнительный анализ трёх СУБД: Redis, PostgreSQL и SQLite с целью определения оптимального хранилища для указанного класса систем.

Основная часть. Разработанная система представляет собой клиент-серверное приложение с использованием протокола WebSocket для доставки обновлений в реальном времени. Серверная часть реализована на языке Go с применением принципов чистой архитектуры, что позволяет подключать различные адаптеры хранения без изменения бизнес-логики. Клиентская часть использует Vue.js и аппаратно-ускоренный рендеринг WebGL для отображения полотна размером 500 × 250 пикселей. Инфраструктура контейнеризована средствами Docker Compose и включает подсистему мониторинга на основе Prometheus и Grafana.

Рабочая нагрузка системы формализована в виде трёх операций: (1) запись единичного пикселя с сохранением полной истории изменений координаты; (2) чтение последнего состояния отдельного пикселя; (3) пакетное чтение всего полотна при подключении нового клиента. Для каждой из рассматриваемых СУБД спроектирована схема хранения, учитывающая специфику модели данных: key-value списки в Redis, реляционные таблицы в PostgreSQL и SQLite [2, 3].

Сравнительная оценка СУБД проводилась на двух уровнях: на уровне хранилища и на уровне приложения под нагрузкой. На уровне хранилища для каждой СУБД выполнялись пять сценариев: (1) последовательная запись пикселей (инициализация полотна); (2) конкурентная запись при нескольких параллельных потоках; (3) пакетное чтение текущего состояния полотна; (4) смешанная нагрузка 80 % записей и 20 % чтений; (5) рост истории — многократная запись в одни и те же координаты для оценки деградации при увеличении объёма данных. Измерялись пропускная способность (операций в секунду), задержки на перцентилях, а также количество ошибок. Нагрузочное тестирование приложения выполнялось инструментом k6 по протоколу WebSocket при трёх уровнях интенсивности. Сбор метрик в ходе бенчмарков и продуктивной эксплуатации осуществлялся подсистемой Prometheus; визуализация и сравнение результатов — в Grafana.

Система была развёрнута в продуктивной среде в рамках университетского мероприятия. За 5 дней работы в системе было зарегистрировано 195 пользователей пяти мегафакультетов, совершивших 88 333 размещений пикселей на полотне из 125 000 координат. Подсистема мониторинга на основе Prometheus подтвердила стабильность временных характеристик Redis под реальной нагрузкой: медианная задержка обработки WebSocket-сообщения не превысила допустимых значений при пиковой интенсивности ~65 сообщений/мин.

Выводы. Проведённый сравнительный анализ показал, что для систем совместного редактирования в реальном времени с аппендитивным характером записи и требованием низкой задержки чтения Redis является оптимальным решением по совокупности критериев:

пропускная способность, задержка чтения и простота эксплуатации. PostgreSQL обеспечивает ACID-гарантии, избыточные для данного класса задач при единственном экземпляре сервера, ценой существенного снижения производительности записи. SQLite демонстрирует высокую скорость записи в однопоточном режиме, но не поддерживает конкурентный доступ, необходимый для многопользовательского приложения. Результаты продуктивного развёртывания подтвердили обоснованность выбора Redis.

Список использованных источников:

1. Redis: документация [Электронный ресурс] // Redis. – URL: https://redis.io/docs/ (дата обращения: 12.02.2025).
2. PostgreSQL Documentation [Электронный ресурс] // The PostgreSQL Global Development Group. – URL: https://www.postgresql.org/docs/ (дата обращения: 12.02.2025).
3. SQLite Documentation [Электронный ресурс] // SQLite. – URL: https://www.sqlite.org/docs.html (дата обращения: 12.02.2025).
4. The Go Programming Language. Documentation [Электронный ресурс] // Go. – URL: https://go.dev/doc/ (дата обращения: 12.02.2025).
5. Vue.js — The Progressive JavaScript Framework. Documentation [Электронный ресурс] // Vue. – URL: https://vuejs.org/ (дата обращения: 12.02.2025).
6. The WebSocket API (WebSockets) [Электронный ресурс] // MDN Web Docs. – URL: https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API (дата обращения: 12.02.2025).
7. The WebSocket Protocol. RFC 6455 [Электронный ресурс] // IETF. – 2011. – URL: https://www.rfc-editor.org/rfc/rfc6455 (дата обращения: 12.02.2025).
8. Prometheus. Monitoring system and time series database. Documentation [Электронный ресурс] // Prometheus. – URL: https://prometheus.io/docs/ (дата обращения: 12.02.2025).
9. Grafana documentation [Электронный ресурс] // Grafana Labs. – URL: https://grafana.com/docs/grafana/latest/ (дата обращения: 12.02.2025).
10. Grafana k6. Open source load testing. Documentation [Электронный ресурс] // Grafana. – URL: https://k6.io/docs/ (дата обращения: 12.02.2025).
11. Docker documentation [Электронный ресурс] // Docker. – URL: https://docs.docker.com/ (дата обращения: 12.02.2025).
12. Wang Z., Luo Z., Jin J. и др. Are We Testing the Right Thing? On the Sensitivity of DBMS Performance to Experimental Settings // Proceedings of the VLDB Endowment. – 2022. – Т. 15. – С. 1439–1452.
13. Taipalus T. A systematic literature review on database management system performance comparisons // Software Quality Journal. – 2023. – Т. 31. – С. 1–36.

Автор _____ Бессонов В.Ю.

Научный руководитель _____ Жданов А.Д.