

УДК 004.852

**ПОДХОДЫ К ХРАНЕНИЮ ТЕЛЕМЕТРИИ КАРЬЕРНЫХ  
САМОСВАЛОВ**

Орлов И.А., студент гр. ИТб-211, IV курс

Научный руководитель: Сыркин И.С., доцент (к.н.) кафедры ИиАПС

Кузбасский государственный технический университет

имени Т.Ф. Горбачева, г. Кемерово

Система с высокой частотой приемки сообщений и большим набором параметров порождает огромное количество данных, а также особые требования к БД. Такая система хранения должна легко масштабироваться под растущий объем данных и обеспечивать запись без потери информации. Одновременно с этим необходима эффективная выборка данных в реальном времени для мониторинга и анализа посредством визуализации и построения графиков на веб-интерфейсе. Такие запросы включают получение последних значений, а также агрегацию по временным интервалам. Не стоит забывать и про требования к нормализации БД для оптимального распределения информации и более удобной структуры.

В качестве возможных решений можно использовать следующие технологии: NoSQL (MongoDB, Redis и др.) или SQL (PostgreSQL+TimescaleDb и другие реляционные решения).

MongoDB – документо-ориентированная СУБД, хранящая данные в формате JSON для неструктурированных данных. Данные хранятся в формате «ключ-значение», то есть: [{objId: ObjectId("1234"), model: 'G456'}, { objId: ObjectId("slovo"), year: 2003}]. Основные преимущества и недостатки MongoDB [4]. Преимущества:

1. Гибкая схема хранения данных позволяет легко изменять содержание посредством добавления новых полей. Также данные в таких СУБД могут иметь разное содержание, поля могут отсутствовать или быть расположены в разном порядке [1].

2. Возможность сохранения вложенных конструкций, например координаты локации можно легко записать в виде - "location\_coords": {"lat": 55.751244, "lon": 37.618423}.

3. Репликация и горизонтальная масштабируемость, позволяют легко обрабатывать и анализировать большие объемы информации.

Недостатки:

1. Для хранения индексов MongoDB использует оперативную память и в случае, если индексы занимают большее количество, чем памяти доступно в памяти, то скорость запросов падает.
2. Повторяющиеся ключи увеличивают объем данных, в случае внесения данных все статические данные самосвала будут повторяться, включая его модель и тд.
3. В случае длительного хранения данных скорость запросов падает, так как приходится обходить большое количество записей.

На основании вышеперечисленного можно сказать следующее, MongoDB не подходит для долгосрочного хранения телеметрии и для быстрой выборки данных для аналитики, что критически важно для отслеживания работы карьера.

В качестве SQL-решения для хранения телеметрии карьерных самосвалов рассмотрим PostgreSQL и PostgreSQL с дополнением TimescaleDB.

PostgreSQL – реляционная база данных, которая позволяет четко структурировать данные, а также выполнять сложные аналитические запросы. Однако с ростом объема производительность работы снижается, а также отсутствует автоматическое разделение по временным рядам [2]. Разберем основные проблемы:

1. При добавление новых данных с временными метками, СУБД сканирует сразу все строки, что сильно увеличивает время запроса.
2. Отсутствие автоматического разделения данных. Разбиение данных на части администратор должен осуществлять вручную.
3. Удаление данных происходит построчно, что достаточно медленно в условиях больших данных.

TimescaleDB дополнение для PostgreSQL, которое вносит концепцию гипертаблицы – виртуальная таблица, которая разграничивает данные по временным интервалам, а также вносит инструментарий для работы с ними. Каждый такой интервал(чанк) является внутренней таблицей PostgreSQL, хранящей данные за определенный диапазон времени для одного или нескольких объектов, который отображается для пользователя в виде единой таблицы. Также ускорена работа удаления данных посредством удаления чанков, а не только строк. Помимо разбиения на чанки вводятся новые функции, позволяющие работать с временными метками, которые содержат данные телеметрии, например:

1. Функция «time\_bucket» позволяет ранжировать и группировать данные по временным интервалам (минуты, часы, дни и тд). В случае необходимости, имеется возможность указания дополнительного параметра для группировки, например, гаражный номер самосвала. Пример запроса, в котором

производится агрегация данных для просмотра средней скорости самосвала в зависимости от времени: `SELECT time_bucket('1 hour', timestamp) AS period, truck_id, avg(speed) AS avg_speed FROM telemetry WHERE timestamp > NOW() - INTERVAL '1 day' GROUP BY period, truck_id ORDER BY period.`

2. Функции «first» и «last» находят первое и последнее значение для заданных временных интервалов. Запрос, реализующий выборку данных для определения уровня топлива по часам: `SELECT truck_id, first(fuel_level, timestamp) AS first_flvl, last(fuel_level, timestamp) AS last_flvl FROM telemetry WHERE timestamp > NOW() - INTERVAL '1 day' GROUP BY truck_id.`

3. Функция «time\_bucket\_gapfill» заполняет пропущенные интервалы, в случае, когда в данных есть разрывы (отсутствуют данные за определенное время), эта функция добавляет пустые интервалы. Пример: `SELECT time_bucket_gapfill('1 minute', timestamp) AS period, truck_id, coalesce(avg(speed), 0) AS avg_speed FROM telemetry WHERE timestamp > NOW() - INTERVAL '1 hour' GROUP BY period, truck_id ORDER BY period.` В запросе пропущенные минуты заполняются нулями.

Различия работы классического PostgreSQL и PostgreSQL с дополнением TimeScaleDB:

1. В TimeScaleDB имеется автоматическое партиционирование данных по временным интервалам, вместо ручного администрирования в классическом PostgreSQL, что ускоряет выборку и добавление данных.

2. Ускоренное удаление. При использовании TimescaleDB данные удаляются целыми партициями, а не посточно.

3. Ускоренная группировка. «time\_bucket» группирует быстрее, чем group by в PostgreSQL [3].

Рассмотренные решения показали, как сильные, так и слабые стороны. NoSQL-решения имеют высокую скорость вставки и хорошо справляются с нагрузкой и объемом информации, однако MongoDB не оптимизирован для сложных аналитических запросов и поиск данных по времени, также отсутствует автоматическое партиционирование по временным интервалам, что снижает производительность при больших объемах данных. Реляционная база данных в виде PostgreSQL обеспечивает гибкость запросов и масштабируемость данных, но при больших объемах информации и отсутствии автоматического разделения данных по промежуткам времени производительность PostgreSQL снижается из-за чего необходимо использовать сторонние расширения для ускорения и оптимизации работы. TimescaleDB закрывает эти задачи за счет автоматического секционирования данных по интервалам, распределения по чанкам, методов группировки, повышая скорость работы сложных аналитических запросов.

Таким образом, для системы хранения телеметрии карьерных самосвалов, где важно быстрое действие и аналитическая гибкость, TimescaleDB предоставляет

оптимальное решение, сочетая в себе богатый SQL-инструментарий, надежное хранение данных, дополнения для улучшения работы с временными рядами.

### Список литературы:

1. MongoDB Documentation. [Электронный ресурс]. – Режим доступа: URL: [Data Modeling - MongoDB Manual v8.0 - MongoDB Docs](#) (дата обращения: 26.03.2025).
2. PostgreSQL Documentation. Introduction to PostgreSQL. [Электронный ресурс]. – Режим доступа: URL: <https://www.postgresql.org/docs/> (дата обращения: 26.03.2025).
3. TimescaleDB Documentation. Getting Started with TimescaleDB. [Электронный ресурс]. – Режим доступа: URL: <https://www.timescale.com/docs> (дата обращения: 26.03.2025).
4. Бегунов, И. В. Введение в MongoDB: документационные практики и решения для бизнеса / И. В. Бегунов, А. И. Смирнов // Программирование и базы данных. – 2019. – С. 27-34.