

Научная статья

DOI: 10.15593/2224-9397/2024.2.08

УДК 004.652

**И.А. Шмидт**Пермский национальный исследовательский политехнический университет,  
Пермь, Российская Федерация**СИСТЕМА УПРАВЛЕНИЯ БАЗОЙ ДАННЫХ ДЛЯ РАБОТЫ  
С ДАННЫМИ СТЕНДОВЫХ ИСПЫТАНИЙ  
СЛОЖНЫХ ИЗДЕЛИЙ**

Исследуются модели хранения временных рядов стендовых испытаний высокотехнологичных производств. В ходе эксплуатации испытательные стенды генерируют поток значений измеренных параметров, которые подлежат хранению с последующей обработкой. Как правило, способ организации хранения таких данных не позволяет корректно использовать данные после проведения испытаний, а объёмы этих данных могут достигать сотен гигабайт. В статье рассматриваются модели хранения данных с использованием реляционной, колоночной и документоориентированной систем управления базами данных. Проведено тестирование предложенных моделей на скорость доступа к данным, представляющим временные ряды, характерные для стендовых испытаний. Наборы тестовых данных для тестирования баз данных, основанных на различных моделях, были подготовлены с учетом специфики архитектуры выбранной базы данных. Показано, что данные стендовых испытаний, несмотря на внешнее сходство, существенно отличаются по структуре и предъявляемым требованиям от данных, генерируемыми устройствами интернета вещей. **Цель исследования:** заключается в определении оптимального метода для хранения больших объемов данных, представляющих результаты испытаний сложных изделий. **Методы:** в рамках исследования были спроектированы схемы данных, представляющие обобщенные результаты стендовых испытаний, и проведено тестирование выбранных систем управления базами данных на скорость доступа. **Результаты:** полученные результаты демонстрируют, что при применении специально спроектированных схем данных стендовых испытаний модельные эксперименты, системы хранения, основанные на документоориентированной архитектуре, дают на порядок лучшую эффективность, чем другие способы доступа. **Практическое значение:** результаты исследования могут найти применение при проектировании систем хранения данных для результатов стендовых испытаний сложных изделий, особенно при разработке новых изделий.

**Ключевые слова:** временные ряды, данные стендовых испытаний, NoSQL, MongoDB.

I.A. Shmidt

Perm National Research Polytechnic University, Perm, Russian Federation

## **DATABASE MANAGEMENT SYSTEM FOR WORKING WITH DATA FROM BENCH TESTS OF COMPLEX PRODUCTS**

The article examines models for storing time-series data from high-tech industry bench tests. During operation, test benches generate a large volume of parameter values that need to be stored and processed. Typically, the way this data is stored makes it difficult to use after testing, as the volume can reach hundreds of gigabytes. The article discusses three data storage models: relational database management systems, column-based database management systems, and document-oriented database management systems. These models were tested to determine their speed of accessing data representative of typical bench test time series. To test the models, data sets were prepared based on the specific architecture of each Database Management Systems. It was found that bench test data differs significantly from data generated by IoT devices in terms of structure and requirements. **The aim of the research:** The main objective of the study is to determine the most effective method for storing large volumes of data that represent the test results of complex products. **Methods:** As part of the research, data models representing generalized test results were developed, and Database Management Systems were tested to assess their ability to access data quickly. **Results:** The findings indicate that when utilizing specially designed data structures for bench testing, document-oriented databases provide significantly better performance in terms of efficiency compared to other access methods, as demonstrated by model experiments. **Practical implications:** The results of this research can be applied in the design of data storage solutions for the test results from bench testing of complex products, particularly in the development of novel products.

**Keywords:** time series, bench test data, NoSQL, MongoDB.

### **Введение**

Стендовые испытания – это один из важнейших этапов жизненного цикла разработки и производства высокотехнологичных изделий. Количество испытательных стендов на современном предприятии может измеряться десятками, а в исключительных случаях сотнями, при этом на каждом из стендов проводятся тысячи испытаний. Каждое испытание изделия на стенде сопровождается фиксацией большого (до нескольких тысяч) количества параметров, привязанных к временным отметкам, т.е. объектом хранения будет временной ряд. Количество временных отметок при проведении испытания может достигать нескольких миллионов.

Как правило, первоначально регистрация параметров в темпе эксперимента осуществляется путём записи их значений на локальный диск, чаще всего непосредственно в двоичном виде. При этом если на стенде установлено несколько систем регистрации для различных групп параметров, то для каждого испытания будет создано несколько

файлов специфического формата. Кроме непосредственно данных измерений при испытании могут создаваться файлы, содержащие дополнительные сведения: об условиях испытания, о стенде, об измеряемых параметрах и каналах измерения и т.д.

В дальнейшем файлы, содержащие всю информацию об испытании, обычно копируются в единое файловое хранилище, эта операция осуществляется вручную специалистами, ответственными за проведение данного испытания. Для упорядочивания полученной при испытаниях информации обычно предпринимается попытка структурировать данные путем размещения исходных файлов в каталогах с формализованным наименованием, но структура хранения, как правило, получается нечёткой вследствие ручного размещения данных в файловой структуре, т.е. из-за присутствия человеческого фактора. Пользователи при необходимости находят данные по испытаниям только на основании соглашений по структуре файловых каталогов и наименований файлов. Человеческий фактор при размещении данных приводит к тому, что могут быть нарушены установленная структура каталогов и правила именования файлов, доступ к данным испытаний будет затруднен.

Ситуация отягощается тем, что результаты испытаний на разных стендах регистрируются различными унаследованными системами, каждая из которых хранит данные в файлах своего собственного формата. Разработать единую систему обработки данных при файловом способе хранения практически невозможно.

В [1] предлагается универсальное решение для регистрации, хранения, поиска и извлечения больших объемов данных на предприятиях, осуществляющих множественные измерения параметров при стендовых испытаниях. Данные измерений по-прежнему хранятся в виде файлов, но формат их унифицирован.

Работа [2] также поднимает проблему, связанную со значительными затратами на хранение данных, полученных в результате мониторинга промышленных систем. Предлагается иерархическая архитектура для хранения данных временных рядов. Оба эти решения основаны на едином способе регистрации параметров и не годятся для предприятия, имеющего набор различных унаследованных систем регистрации.

Радикальным выходом является хранение данных об испытаниях в единой базе данных. После первичной регистрации параметров вся

информация экспортируется в базу данных, которая обеспечивает ее хранение в едином формате и извлечение по запросам. СУБД, предназначенная для работы таких промышленных данных, должна обеспечивать как эффективный ввод данных, так и их извлечение, что является сложной задачей. Связанной задачей является организация информации, представляющей временные ряды, в базе данных. Способ представления данных является основным фактором, определяющим эффективность всей системы.

Многие исследователи прилагают значительные усилия к разработке методов, позволяющих оптимизировать хранение данных временных рядов. Исходные временные ряды генерируются или в результате мониторинга промышленных систем (включая измерения при испытаниях изделий), или как результат фиксации потока данных от устройств интернета вещей. В [3] рассматриваются работы с временными рядами измерений с точки зрения баз данных. Рассматриваются различные аспекты применения для хранения и извлечения такого типа данных, как традиционные реляционные СУБД, так и решения, классифицируемые как NoSQL-системы. По мнению автора, из последнего класса предпочтительнее выбрать БД Cassandra (по состоянию на 2017 г.).

В статье [4] приводятся результаты сравнительного анализа различных типов СУБД в контексте наиболее критических задач, производимых с данными временных рядов, при этом помимо реляционных СУБД рассматриваются возможности графовых и циклических СУБД.

Исследование [5] представляет результаты эмпирического сравнения трех NoSQL СУБД, а именно: Cassandra, MongoDB и InfluxDB, в поддержании и извлечении гигабайт реальных данных IoT. В тестах для конкретного набора данных временных рядов наилучшие результаты продемонстрировала InfluxDB.

В [6] также проводилось тестирование различных СУБД для данных мониторинга технической системы, которая сочетает в себе датчики реального времени и механизм обработки данных для решения конкретных задач, таких как обнаружение аномалий, прогнозирование и интерпретация данных. Авторы делают вывод, что колоночная СУБД ClickHouse, разработанная для задач веб-аналитики, является лучшим решением для хранения данных временных рядов.

В работе [7] для обработки данных временных рядов, сгенерированных техническими системами, также оцениваются различные СУБД, помимо классических реляционных моделей и баз данных NoSQL исследуется недавно появившаяся группа – базы данных NewSQL, т.е. реляционные СУБД, изначально построенные под требования горизонтальной масштабируемости, в которых увеличение производительности системы достигается ценой использования большого количества вычислительных ресурсов.

Методика тестирования баз данных временных рядов применительно к системе мониторинга клиентской IT-инфраструктуры рассмотрена в [8]. В этой работе детализированы критерии сравнения СУБД и описана структура хранимых данных. Так же, как во всех вышеперечисленных работах, исследования проводились, по сути, только для одной таблицы (записываются только таймсерии зафиксированных параметров). Для тестирования каждой из СУБД таблица, состоящая из большого числа записей, загружается в каждую из них, а затем выполняется операция чтения.

Такая простейшая модель хранимых данных не подходит для задачи хранения результатов стендовых испытаний, так как каждый стенд производит измерения своего собственного множества параметров (которые считываются с разными темпами), при этом даже на одном и том же стенде при разных испытаниях фиксируются различные наборы параметров. Кроме этого в условиях реального испытательного производства данные временных рядов являются важной, но не единственной информацией, описывающей испытания, также необходимо хранить как атрибуты самого испытания, так и атрибуты параметров и каналов измерения.

В серии работ [9–11] предложена схема хранения данных временных рядов, порожденных стендовыми испытаниями, с использованием документоориентированной СУБД MongoDB и рассмотрены вопросы внутренней организации данных в такой базе данных.

В отличие от всех рассмотренных выше работ, в которых массив измерений при испытаниях изделий представляется просто таблицей с временными метками, в нашем исследовании будет выполняться сравнительный анализ применимости данных о стендовых испытаниях различных типов СУБД с учетом различия их способа представления информации.

## **Структура исходной информации**

Испытание – это корневое понятие, вокруг которого собираются все остальные данные. ГОСТ 16504 определяет испытание как экспериментальное определение количественных и качественных характеристик параметров изделия путём воздействия на него спланированного комплекса воздействий [12].

Сущность испытания заключается в регистрации параметров. Параметр описывается именем и атрибутами. Атрибуты параметра описывают его дополнительные свойства, например: единица измерения, номер (название) канала измерения, информация о датчике, его калибровке и пр. Каждое испытание характеризуется своим набором регистрируемых параметров.

Набор значений какого-либо параметра, зарегистрированных при проведении испытания, привязанных ко времени, будем называть «трендом». Для пользователя данные испытания представляются в виде набора трендов, т.е. таблицы, заголовками столбцов которой являются имена параметров. Отдельным видом параметра является «параметр с отметками времени». Значения этого параметра сами по себе не несут информации о характеристиках, полученных при испытании, а служат для привязки других значений параметров ко времени, когда они были получены. Отметки времени являются сквозными для всех трендов испытания. Набор значений параметров, привязанных к одной отметке времени, называется «замер». В одном замере может присутствовать от одного значения параметра до всего набора параметров испытания, т.е. часть атрибутов в строке может быть пустой, что практически означает, что при разных замерах могут регистрироваться различные наборы параметров. Таким образом, основной объем данных будет иметь вид таблицы, у которой количество столбцов и их состав будут специфичны для каждого испытания, а количество строк соответствует количеству замеров.

## **Обзор методов хранения данных**

Однозначной классификации типов СУБД до сих пор не существует, в разные десятилетия эта классификация выглядела по-разному. Каждое появление СУБД, использующей новые подходы, меняет подход к классификации. Старая терминология быстро устаревает

и вытесняется новой. Одна из наиболее универсальных классификаций, актуальных на сегодня, приведена в [13].

В дальнейшем будут рассматриваться только типы СУБД, потенциально пригодные для хранения данных временных рядов. Будем выделять традиционные реляционные СУБД и NoSQL СУБД. Среди NoSQL СУБД будем рассматривать документоориентированные СУБД. Особое место в классификации занимают базы данных временных рядов (TSDBMS) [14]. Это особое место объясняется тем, что прочие виды БД выделяются в классификации по способу организации данных, а базы данных временных рядов просто определяются как система, оптимизированная для хранения и обслуживания временных рядов и использующая временные метки, т.е. по способу применения. Другое название таких СУБД – темпоральные базы данных. К СУБД временных рядов причисляют такие системы, как: TimescaleDB [15] и QuestDB [16], которые используют реляционную концепцию хранения данных, Prometheus [17] использует модель, «ключ-значение», а Hbase [18], и InfluxDB [19] – это колоночно-ориентированные базы данных.

При выборе для тестирования конкретных СУБД, представляющих свои семейства, принималась во внимание такая характеристика, как связанность (coupling) [20], под которой понимается мера того, насколько взаимозависимы разные подпрограммы или модули. Сильная связанность означает, что для функционирования системы дополнительно требуется множество сторонних библиотек или приложений. Для систем хранения промышленных предприятий требуется выбирать СУБД с минимально возможной связанностью.

Многочисленные сравнения и рейтинги быстродействия БД [21, 22], в которых обычно используют выражения «БД xxxx производит выборку данных в  $n$  раз быстрее, чем БД уууу», малоинформативны, так как обычно сравнение производится на очень простых схемах данных, чаще всего просто сравнивают скорость доступа к одной записи. В работе для тестирования СУБД с различными моделями хранения каждый раз будет применяться схема данных, в наибольшей степени подходящая для выбранной архитектуры СУБД. Сами наборы данных для тестирования будут одинаковыми для всех тестов и представлять обобщенные результаты стендовых испытаний, но их представление будет специфическим для каждого теста.

В данной работе тестовыми данными, предназначенными для извлечения из БД, выступают предварительно сгенерированные наборы временных рядов, представляющих отдельные стендовые испытания. Каждое испытание фиксирует определенный набор параметров (одни и те же параметры могут участвовать в различных испытаниях). Количество замеров в испытаниях варьируется от десятков до нескольких миллионов. Часть значений параметров в каждом замере может оставаться пустой (null value). В каждом наборе параметров один из параметров выделен как временная отметка.

Предлагается реализовать модели хранения данных, которые рассматриваются как альтернативы практически во всех работах, посвященных этой тематике: реляционную, колоночную, документоориентированную. Для всех трех моделей будет произведено тестирование скорости доступа к данным.

### ***Реляционные СУБД***

Основное преимущество использования реляционной модели при хранении и обработке данных временных рядов – это широкое распространение и доступность таких систем, подавляющее большинство крупных корпоративных информационных систем основано на РСУБД. Как показано в работах, упомянутых во введении, реляционные базы данных до сих пор претендуют на роль хранилищ для данных временных рядов. Наиболее часто в качестве РСУБД для хранения временных данных рассматривается MySQL, в данной работе для тестирования реляционной модели также была выбрана эта СУБД.

На первый взгляд набор трендов в представлении для реляционной модели это просто таблица, но как уже было изложено выше, каждое испытание фиксирует свой собственный, уникальный набор параметров (каждый параметр в реляционном представлении это столбец таблицы). Таким образом, «наивный» подход, при котором для каждого нового испытания создается своя отдельная таблица со своей собственной схемой (собственным набором параметров), на практике не применим – трудно представить логику приложения с базой данных с десятками тысяч таблиц. Поэтому было принято решение хранить все значения параметров для разных испытаний в одной таблице. Связь испытания с набором параметров, которые фиксируют значения этих параметров, осуществляется при помощи дополнительной таблицы соединений. Эта сущность обеспечивает связь «Многие – Ко – Многим» между испыта-



нием и параметром (каждый параметр может присутствовать в нескольких испытаниях, но при каждом испытании измеряется и фиксируется несколько параметров). Схема данных представлена на рис. 1.

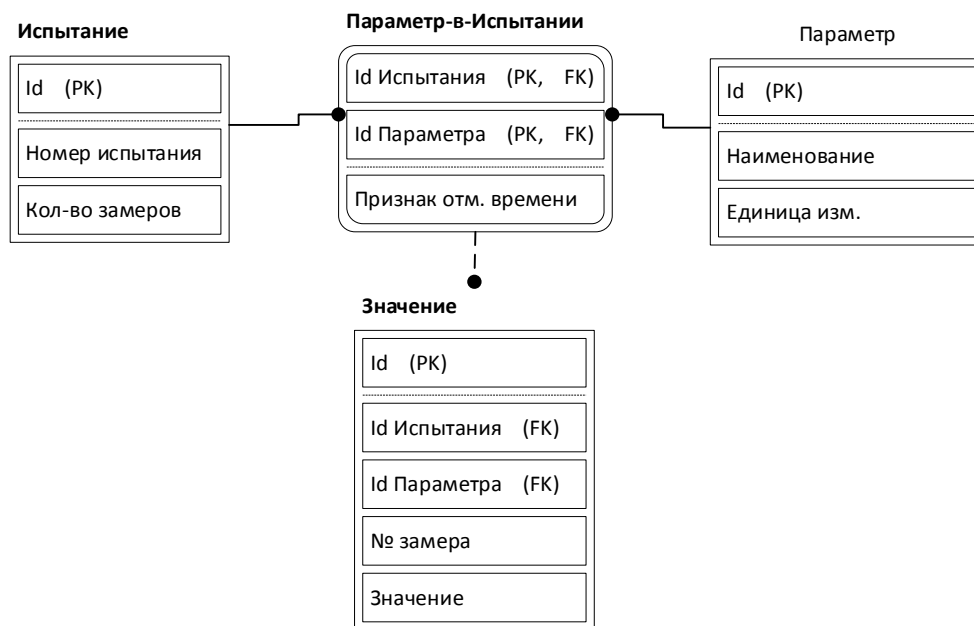


Рис. 1. Представление данных временных рядов стендовых испытаний для реляционной СУБД

Всего в модели четыре сущности:

- испытание – хранит информацию о собственно стендовом испытании, т.е. содержит набор атрибутов, характеризующих конкретное испытание. В представленной схеме приведен минимальный набор атрибутов. На практике их состав может быть расширен, учитывая специфику стенда и вида испытания;

- параметр – хранит информацию о параметрах (каналах измерения), которые могут быть задействованы при проведении испытания. Один из параметров должен являться отметкой времени, что и делает нашу схему пригодной для хранения временных рядов;

- параметр-в-Испытании – вспомогательная, промежуточная сущность, которая определяет состав параметров в каждом конкретном испытании. Она разрешает коллизию «Многие – Ко – Многим»;

- значение – содержит информацию о значении параметра в испытании в момент замера.

Методика тестирования представлена ниже, численные результаты сведены в таблице.

### Колоночные СУБД

Колоночные СУБД работают с колоночными семействами, которые являются аналогами таблиц в РСУБД. Колоночные семейства содержат колонки с записями (т.е. данные хранятся по столбцам), такая организация обеспечивает более эффективный доступ к данным при запросе по сравнению с РСУБД.

В качестве тестируемой колоночной СУБД была выбрана InfluxDB, которая часто присутствует в обзорах. Как уже было изложено выше, базу данных InfluxDB обычно не относят к колоночным СУБД, классифицируя ее как базу данных временных рядов (Time Series Database). СУБД данного типа как раз и были разработаны под хранение данных временных рядов. Основной их особенностью является поддержка высокой скорости приема данных, что наиболее критично для приложений Интернета вещей, но для задач обработки результатов испытаний важнее скорость извлечения информации. С точки зрения архитектуры темпоральные БД могут основываться как на базе РСУБД, так и на базе NoSQL. Идеологически InfluxDB наиболее близка к колоночным СУБД. Колоночное семейство в данном случае именуется измерением (measurement), а колонки называются полями (field). Применительно к нашей задаче измерение, measurement – это испытание, а поле, соответственно, – это тренд.

Данные стендовых испытаний в терминах InfluxDB показаны на рис. 2.

Испытание 1			Испытание М		
Параметр 1	Параметр 2	Параметр К	Параметр 1	Параметр 2	Параметр К
Значение 1	Значение 1	Значение 1	Значение 1	Значение 1	Значение 1
Значение 2	Значение 2	Значение 2	Значение 2	Значение 2	Значение 2
Значение 3	Значение 3	Значение 3	Значение 3	Значение 3	Значение 3
...	...	...	...	...	...
...	...	...	...	...	...
Значение N	Значение N	Значение N	Значение N	Значение N	Значение N

Рис. 2. Представление данных временных рядов стендовых испытаний для колоночной темпоральной СУБД

В отличие от РСУБД, где все таблицы должны соответствовать заданной схеме, колоночные семейства содержат строки, каждая из

которых может иметь собственный формат. Поэтому наличие большого количества семейств не является такой проблемой, как большое количество таблиц в реляционной схеме.

Определенной проблемой является то, что InfluxDB предназначена только для хранения временных рядов и не способна работать с прочей информацией, ассоциированной с испытанием.

### Документоориентированные СУБД

Другим перспективным множеством NoSQL баз данных являются документоориентированные СУБД (Document-oriented database). Такие СУБД ориентированы на хранения иерархических JSON-структур данных – документов, сгруппированных в «коллекции». Эти СУБД отлично сочетают гибкость с горизонтальной масштабируемостью.

В качестве документоориентированной СУБД предлагается использовать одну из наиболее популярных – MongoDB, которая также рассматривается практически во всех обзорах по хранению временных рядов. В упрощенном виде способ организации данных показан на рис. 3.

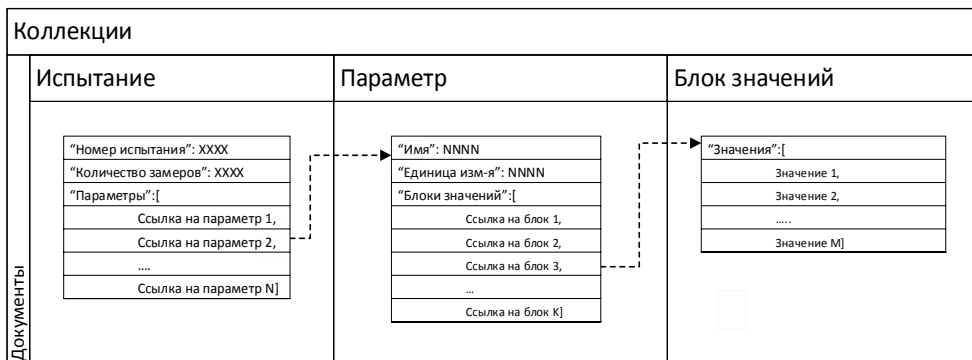


Рис. 3. Представление данных временных рядов стендовых испытаний для колоночной СУБД

Схема данных использует три коллекции.

Коллекция «Испытание» включает документы, каждый из которых описывает конкретное стендовое испытание (представлен минимальный набор атрибутов). Каждый такой документ содержит массив ссылок «Параметры». Ссылки из этого массива указывают на документы из коллекции «Параметр», описывающие информацию о параметре, который был задействован при проведении испытания. Документы из массива «Параметры», в свою очередь, содержат ссылки на документы

из коллекции «Блок значений», содержащие собственно массив измеренных значений. Для извлечения нужного значения требуется по временной метке вычислить два индекса: индекс блока значений и индекс значения в этом блоке. Хранение измеренных значений как элементов массива обеспечивает эффективность потокового чтения из временного ряда. Такая схема эффективна в случае, если временной ряд плотно заполнен данными, т.е. пустое значение (null) встречается редко. Для разреженных временных рядов нужно использовать другую схему, рассмотрение которой выходит за границы данной работы.

### **Тестирование моделей хранения временных рядов**

Для оценки эффективности моделей временных рядов, реализованных в парадигме соответствующих баз данных, проводилось тестирование на скорость доступа к выборкам значений параметров различной длины. Оценивалось среднее время доступа к одному значению параметра. В качестве тестовых данных выступали предварительно сгенерированные временные ряды, имитирующие результаты стендовых испытаний [23] (количество замеров, т.е. строк в исходной таблице, от 10 тыс. до 1 млн, количество параметров от 50 до 500 шт.). Всего было подготовлено девять последовательностей, имитирующих данные типичных стендовых испытаний. Сгенерированные данные представлялись в виде CSV-таблиц, которые экспортировались в каждую из тестируемых баз данных. Таким образом, каждая из СУБД после экспорта содержала одни и те же данные, но в специфическом для каждой из моделей виде.

Тестирование заключалось в извлечении данных из последовательностей по 100 тыс. замеров. Каждый замер определялся парой элементов – номер замера, номер параметра. Выполнялось четыре теста для каждого из временных рядов:

- считывание единичного случайного замера;
- считывание серии из 100 подряд идущих значений, начинающейся от случайного замера;
- считывание серии из 1000 подряд идущих значений, начинающейся от случайного замера;
- считывание серии из 10000 подряд идущих значений, начинающейся от случайного замера.

Измерения проводились при помощи специально разработанного тестового приложения для каждого из девяти стендовых испытаний в каждой СУБД. Измеренные времена доступа для каждого из тестов усреднялись по всем стендовым испытаниям. Таблица представляет полученные усреднённые значения времени доступа к элементам временного ряда, хранящимся в тестируемых базах данных.

### Результаты тестирования СУБД

№ п/п	Тест	Среднее время доступа к 1-й записи		
		реляционная (MySQL), мс	колоночная, темпоральная (InfluxDB), мс	документоориентированная (MongoDB), мс
1	Единичное значение замера, выбранное случайным образом из временного ряда	150	8	1,8
2	100 подряд идущих значений из временного ряда (начало серии определяется случайно)	1,6	0,11	0,02
3	1000 подряд идущих значений (начало серии определяется случайно)	0,28	0,02	0,002
4	10000 подряд идущих значений (начало серии определяется случайно)	0,0002	0,17	0,02

Как и следовало ожидать, извлечение данных длинными сериями происходит на порядки быстрее, чем обращение к одиночным замерам. На практике пользователи почти всегда будут извлекать данные для обработки длинными сериями (интервалами), например, чтобы отобразить их в таблице или на графике.

Результаты тестирования показывают, что реляционные СУБД менее пригодны для хранения временных рядов, представляющих данные о стендовых испытаниях, чем NoSQL базы. Это вполне предсказуемый результат, так как традиционные базы данных изначально плохо совместимы с природой временного ряда. Основой реляционной модели является отношение – неупорядоченное множество кортежей (строк

отношений), тогда как определяющим признаком временного ряда как раз является упорядоченность его элементов.

Значительно более высокая скорость доступа к данным при использовании колоночной СУБД объясняется тем, что происходит обращение сразу к нужной колонке конкретного колоночного семейства, ассоциированного с нужным нам испытанием. При этом системе для выполнения запроса на извлечение требуется перебирать гораздо меньше данных, чем при использовании реляционной модели, где необходимо оперировать строками со всех испытаний.

Использование схемы, ориентированной на СУБД MongoDB, оказывается ещё более эффективной. Высокая эффективность определяется предложенным способом хранения элементов временного ряда. После загрузки документа из коллекции «Испытание» можно сразу получить ссылку на документ «Параметр» с описанием требуемого параметра, который, в свою очередь, имеет ссылки на все массивы измерений значений этого параметра. После чтения требуемого документа «Параметр» задача сводится лишь к получению документа «Блок значений», содержащего массив значений, и получению из массива «Значения» нужного элемента.

Таким образом, такая высокая эффективность документоориентированной схемы объясняется тем, что структура хранения временных рядов изначально спроектирована под требования максимального быстродействия при извлечении данных в отличие от универсальных схем реляционных и колоночных баз данных. В некотором смысле предложенная схема хранения подобна сетевой модели БД, в которой присутствуют указатели, которые соединяют родственную информацию [24]. Так же, как в сетевой модели, за возможность эффективной реализации по показателям быстродействия и использования памяти приходится платить отсутствием универсальности и необходимостью использовать сложный навигационный механизм доступа.

Дополнительным преимуществом докуменориентированной СУБД является то, что помимо собственно данных стендовых замеров в ней можно хранить сопутствующую испытанию дополнительную информацию, такую как: технические отчеты, справки, заключения, акты, протоколы испытаний и т.п. Использование спецификации GridFS [25] позволяет хранить такую информацию непосредственно в БД. Этим обеспечивается высокая скорость доступа и гарантируется

защита этой информации от удаления, перемещения, переименования. Пример хранения паспорта испытания, т.е. документа, содержащего информацию по конкретному испытанию совместно с трендами испытаний, представлен в [26].

### **Заключение**

Несмотря на похожесть данных временных рядов, генерируемых стендовыми испытаниями и Интернетом вещей, к системам, имеющим дело с такими данными, предъявляются совершенно различные требования.

Для задач мониторинга систем, сбора показателей в реальном времени, а также данных Интернета вещей наиболее важны такие показатели, как скорость приема данных и сжатие данных на диске. При этом исходная структура данных фактически сводится к одной таблице, строки в которой добавляются по мере функционирования системы. Для таких условий лучшие результаты демонстрируют СУБД, изначально разработанные для работы с временными рядами [27].

Задачи, связанные с применением временных рядов для хранения результатов стендовых испытаний, напротив, ставят на первое место удобство и скорость извлечения данных. Результаты испытаний записываются в БД однократно, но пользователям постоянно требуются эти данные для анализа результатов. Данные в таких системах имеют более сложную структуру и привязываются к факту проведения испытания технического изделия. Для нашего случая предложенный метод хранения временных данных дает десятикратное преимущество по сравнению с другими методами.

Результаты могут быть использованы для разработки промышленных систем, ориентированных на сбор, хранение и обработку больших объемов сложной технологической информации.

### **Библиографический список**

1. База данных испытаний (БДИ). Комплексное решение по управлению испытаниями, сбору и обработке данных [Электронный ресурс]. – URL: <http://www.nppmera.ru/baza-dannyix-isyitanij> (дата обращения: 22.04.2024).

2. A Hierarchical Storage System for Industrial Time-Series Data / Kevin Villalobos, Víctor Julio Ramírez, Borja Diez, José Miguel Blanco,

Alfredo Goñi, Arantza Illarramendi // IEEE 17th International Conference on Industrial Informatics (INDIN). – 2019.

3. Намиот, Д.Е. Базы данных временных рядов в системах «интернета вещей» / Д.Е. Намиот // Прикладная информатика. – 2017. – Т. 12, № 2 (68). – С. 79–87.

4. Волкова, С.В. Сравнительный анализ возможностей реляционных, графовых и циклических СУБД для хранения и обработки данных временных рядов / С.В. Волкова // Вестник современных исследований. – 2020. – № 1-7 (31). – С. 19–23.

5. Industrial internet of things: persistence for time series with NoSQL databases / Sergio Di Martino, Luca Fiadone, Adriano Peron, Alberto Riccabone, Vincenzo Norman Vitale // IEEE 28th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE). – 2019. – P. 340–345.

6. Rudakov, V. Comparison of time series databases / V. Rudakov, T. Merembayev, Y. Amirgaliyev // 17th International Conference on Electronics Computer and Computation (ICECCO). – IEEE, 2023. – P. 4.

7. A comparison of relational, NoSQL and NewSQL database management systems for the persistence of time series data / C. Praschl, S. Pritz, O. Krauss, M. Harrer // International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME). – IEEE, 2022. – P. 6.

8. Чумак, Д. Как мы тестировали несколько баз данных временных рядов [Электронный ресурс] / Д. Чумак. – URL: <https://www.itsumma.ru/blog/tsdb> (дата обращения: 22.04.2024).

9. Шмидт, И.А. Автоматизация постэкспериментальной обработки и хранения данных стендовых испытаний / И.А. Шмидт, А.С. Желтышев // Вестник Пермского национального исследовательского политехнического университета. Электротехника, информационные технологии, системы управления. – 2020. – № 35. – С. 102–118.

10. Шмидт, И.А. Разработка системы хранения временных рядов в документо-ориентированной базе данных / И.А. Шмидт, А.Б. Петроченков, Д.А. Даденков // Информационно-измерительные и управляющие системы. – 2019. – Т. 17, № 4. – С. 5–11.

11. Оптимизация хранения временных рядов в документоориентированных базах данных / И.А. Шмидт, Д.Р. Жуков, Д.Ю. Лузянин, И.А. Попов, А.Ю. Тимков // Инновационные технологии: теория, ин-



струменты, практика: материалы XIV Междунар. интернет-конф. молодых ученых, аспирантов, студентов (14 ноября – 31 декабря 2022 г.). – Пермь, 2022. – Т. 1. – С. 26–32.

12. ГОСТ 16504-81. Система государственных испытаний продукции. Испытания и контроль качества продукции. Основные термины и определения. – М.: Стандартинформ, 2011.

13. Kosolapov, K. Виды баз данных. Большой обзор типов СУБД [Электронный ресурс]. – URL: <https://habr.com/ru/companies/amvera/articles/754702/> (дата обращения: 22.04.2024).

14. Exact Discovery of Time Series Motifs / Mueen Abdullah, Keogh Eamonn, Zhu Qiang, Cash Sydney, Westover Brandon. – University of California, Riverside, 2009. – P. 473–484. DOI: 10.1137/1.9781611972795.41

15. PostgreSQL ++ for time series and events [Электронный ресурс]. – URL: <https://www.timescale.com/> (дата обращения: 22.04.2024).

16. High performance time series database [Электронный ресурс]. – URL: <https://questdb.io/> (дата обращения: 22.04.2024).

17. From metrics to insight Power your metrics and alerting with the leading open-source monitoring solution [Электронный ресурс]. – URL: <https://prometheus.io/> (дата обращения: 22.04.2024).

18. Apache HBase – Apache HBase™ Home [Электронный ресурс]. – URL: <https://hbase.apache.org/> (дата обращения: 22.04.2024).

19. InfluxDB Time Series Data Platform | InfluxData [Электронный ресурс]. – URL: <https://influxdata.com/> (дата обращения: 22.04.2024).

20. Макконнелл, Стив. Совершенный код / Стив Макконнелл. – 2-е изд. – М.: Русская редакция, 2010. – 896 с.

21. Best Real-Time Databases for 2024 [Электронный ресурс]. – URL: [https://translated.turbopages.org/proxy\\_u/en-ru.ru.5c8cae6f-663e0387-c7fef104-74722d776562/https/www.geeksforgeeks.org/real-time-databases/](https://translated.turbopages.org/proxy_u/en-ru.ru.5c8cae6f-663e0387-c7fef104-74722d776562/https/www.geeksforgeeks.org/real-time-databases/) (дата обращения: 22.04.2024).

22. Драч, В.Е. Анализ популярных нереляционных систем управления базами данных / В.Е. Драч, В.Ю. Ильичев, А.В. Родионов // Системный администратор. – 2022. – № 3 (232). – С. 84–88.

23. Генерация разнотемповых временных рядов / И.А. Шмидт, Д.Р. Жуков, Д.Ю. Лузянин, И.А. Попов, А.Ю. Тимков // Инновационные технологии: теория, инструменты, практика: материалы XIV Междунар. интернет-конф. молодых ученых, аспирантов, студентов (14 ноября – 31 декабря 2022 г.). – Пермь, 2022. – Т. 1. – С. 21–25.

24. Дейт, К.Дж. Введение в системы баз данных / К.Дж. Дейт. – 8-е изд. – М.: Вильямс, 2006. – С. 1328.

25. GridFS – MongoDB Manual v7.0 [Электронный ресурс]. – URL: <https://www.mongodb.com/docs/manual/core/gridfs/> (дата обращения: 22.04.2024).

26. Шмидт, И.А. Разработка оптимального метода хранения временных рядов в документоориентированной базе данных / И.А. Шмидт, А.П. Попов // *Инновационные технологии: теория, инструменты, практика*. – Пермь, 2018. – Т. 1. – С. 233–238.

27. InfluxDB is 5x Faster vs. MongoDB for Time Series Workloads/By Chris Churilo / Nov 17, 2022. – URL: <https://www.influxdata.com/blog/InfluxDB-is-27x-faster-vs-mongodb-for-time-series-workloads/> (дата обращения: 22.04.2024).

## References

1. Baza dannykh ispytaniy (BDI). Kompleksnoe reshenie po upravleniiu ispytaniyami, sboru i obrabotke dannykh [Test Database (DBI). A comprehensive solution for test management, data collection and processing], available at: <http://www.nppmera.ru/baza-dannyix-isytyaniy> (accessed 22 April 2024).

2. Kevin Villalobos, Victor Julio Ramirez, Borja Diez, José Miguel Blanco, Alfredo Goñi, Arantza Illarramendi. A Hierarchical Storage System for Industrial Time-Series Data. *IEEE 17th International Conference on Industrial Informatics (INDIN)*, 2019.

3. Namiot D.E. Bazy dannykh vremennykh riadov v sistemakh “interneta veshchei” [Time series databases in "Internet of Things" systems]. *Prikladnaia informatika*, 2017, vol. 12, no. 2 (68), pp. 79-87.

4. Volkova S.V. Sravnitel'nyi analiz vozmozhnostei relatsionnykh, grafovyykh i tsiklicheskikh SUBD dlia khraneniia i obrabotki dannykh vremennykh riadov [Comparative analysis of the capabilities of relational, graph and cyclic DBMS for storing and processing time series data]. *Vestnik sovremennykh issledovaniy*, 2020, no. 1-7 (31), pp. 19-23.

5. Sergio Di Martino, Luca Fiadone, Adriano Peron, Alberto Riccabone, Vincenzo Norman Vitale. Industrial internet of things: persistence for time series with NoSQL databases. *IEEE 28th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, 2019, pp. 340-345.

6. Rudakov V., Merembayev T., Amirgaliyev Y. Comparison of time series databases. *17th International Conference on Electronics Computer and Computation (ICECCO)*. IEEE, 2023, 4 p.

7. Praschl C., Pritz S., Krauss O., Harrer M. A comparison of relational, NoSQL and NewSQL database management systems for the persistence of time series data. *International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*. IEEE, 2022, 6 p.

8. Chumak D. Kak my testirovali neskol'ko baz dannykh vremennykh riadov [How we tested several time series databases]. available at: <https://www.itsumma.ru/blog/tsdb> (accessed 22 April 2024).

9. Shmidt I.A., Zheltyshev A.S. Avtomatizatsiia posteksperimental'noi obrabotki i khraneniia dannykh stendovykh ispytaniy [Automation of post-experimental processing and data storage of bench tests]. *Vestnik Permskogo natsional'nogo issledovatel'skogo politekhnicheskogo universiteta. Elektrotehnika, informatsionnye tekhnologii, sistemy upravleniia*, 2020, no. 35, pp. 102-118.

10. Shmidt I.A., Petrochenkov A.B., Dadenkov D.A. Razrabotka sistemy khraneniia vremennykh riadov v dokumento-orientirovannoi baze dannykh [Development of a time series storage system in a document-oriented database]. *Informatsionno-izmeritel'nye i upravliaiushchie sistemy*, 2019, vol. 17, no. 4, pp. 5-11.

11. Shmidt I.A., Zhukov D.R., Luzianin D.Iu., Popov I.A., Timkov A.Iu. Optimizatsiia khraneniia vremennykh riadov v dokumentoorientirovannykh bazakh dannykh [Optimization of time series storage in document-oriented databases]. *Innovatsionnye tekhnologii: teoriia, instrumenty, praktika. Materialy XIV Mezhdunarodnoi internet-konferentsii molodykh uchennykh, aspirantov, studentov (14 November - 31 December 2022)*. Perm', 2022, vol. 1, pp. 26-32.

12. GOST 16504-81. Sistema gosudarstvennykh ispytaniy produktsii. Ispytaniia i kontrol' kachestva produktsii. Osnovnye terminy i opredeleniia [State Standard 16504-81. The state system of testing products. Product test and quality inspection. General terms and definitions]. Moscow: Standartinform, 2011.

13. Kosolapov K. Vidy baz dannykh. Bol'shoi obzor tipov СУБД [Types of databases. A large overview of DBMS types], available at: <https://habr.com/ru/companies/amvera/articles/754702/> (accessed 22 April 2024).

14. Mueen Abdullah, Keogh Eamonn, Zhu Qiang, Cash Sydney, Westover Brandon. Exact Discovery of Time Series Motifs. University of California, Riverside, 2009, pp. 473-484. DOI: 10.1137/1.9781611972795.41

15. PostgreSQL ++ for time series and events, available at: <https://www.timescale.com/> (accessed 22 April 2024).

16. High performance time series database, available at: <https://questdb.io/> (accessed 22 April 2024).

17. From metrics to insight Power your metrics and alerting with the leading open-source monitoring solution, available at: <https://prometheus.io/> (accessed 22 April 2024).

18. Apache HBase - Apache HBase™ Home, available at: <https://hbase.apache.org/> (accessed 22 April 2024).

19. InfluxDB Time Series Data Platform | InfluxData, available at: <https://influxdata.com/> (accessed 22 April 2024).

20. Makkonnell Stiv. Sovershennyi kod [Code complete]. 2nd ed. Moscow: Russkaia redaktsiia, 2010, 896 p.

21. Best Real-Time Databases for 2024, available at: [https://translated.turbopages.org/proxy\\_u/en-ru.ru.5c8cae6f-663e0387-c7fef104-74722d776562/https/www.geeksforgeeks.org/real-time-databases/](https://translated.turbopages.org/proxy_u/en-ru.ru.5c8cae6f-663e0387-c7fef104-74722d776562/https/www.geeksforgeeks.org/real-time-databases/) (accessed 22 April 2024).

22. Drach V.E., Il'ichev V.Iu., Rodionov A.V. Analiz populiarnykh nereliatsionnykh sistem upravleniia bazami dannykh [Analysis of popular relational database management systems]. *Sistemnyi administrator*, 2022, no. 3 (232), pp. 84-88.

23. Shmidt I.A., Zhukov D.R., Luzianin D.Iu., Popov I.A., Timkov A.Iu. Generatsiia raznotempovykh vremennykh riadov [Generation of multi-time series]. *Innovatsionnye tekhnologii: teoriia, instrumenty, praktika. Materialy XIV Mezhdunarodnoi internet-konferentsii molodykh uchenykh, aspirantov, studentov (14 November - 31 December 2022)*. Perm', 2022, vol. 1, pp. 21-25.

24. Deit K.Dzh. Vvedenie v sistemy baz dannykh [An Introduction to Database Systems]. 8nd ed. Moscow: Vil'iams, 2006, 1328 p.

25. GridFS - MongoDB Manual v7.0, available at: <https://www.mongodb.com/docs/manual/core/gridfs/> (accessed 22 April 2024).

26. Shmidt I.A., Popov A.P. Razrabotka optimal'nogo metoda khraneniia vremennykh riadov v dokumentoorientirovannoi baze dannykh [Development of an optimal method for storing time series in a document-

oriented database]. *Innovatsionnye tekhnologii: teoriia, instrumenty, praktika. Materialy XIV Mezhdunarodnoi internet-konferentsii molodykh uchenykh, aspirantov, studentov (14 November - 31 December 2018)*. Perm', 2018, vol. 1, pp. 233-238.

27. InfluxDB is 5x Faster vs. MongoDB for Time Series Workloads/By Chris Churilo / Nov 17, 2022, available at: <https://www.influxdata.com/blog/InfluxDB-is-27x-faster-vs-mongodb-for-time-series-workloads/> (accessed 22 April 2024).

### Сведения об авторе

**Шмидт Игорь Альбертович** (Пермь, Российская Федерация) – кандидат технических наук, доцент кафедры «Микропроцессорные средства автоматизации» Пермского национального исследовательского политехнического университета (614990, Пермь, Комсомольский пр., 29, e-mail: [shmidt@pstu.ru](mailto:shmidt@pstu.ru)).

### About the author

**Igor A. Shmidt** (Perm, Russian Federation) – Ph. D., Associate Professor of the Department of Microprocessor Automation of Perm National Research Polytechnic University (614990, Perm, 29, Komsomolsky pr., e-mail: [shmidt@pstu.ru](mailto:shmidt@pstu.ru)).

Поступила: 10.04.2024. Одобрена: 03.06.2024. Принята к публикации: 08.07.2024.

**Финансирование.** Исследование не имело спонсорской поддержки.

**Конфликт интересов.** Автор заявляет об отсутствии конфликта интересов по отношению к статье.

**Вклад** 100 %.

Просьба ссылаться на эту статью в русскоязычных источниках следующим образом:  
Шмидт, И.А. Система управления базой данных для работы с данными стендовых испытаний сложных изделий / И.А. Шмидт // Вестник Пермского национального исследовательского политехнического университета. Электротехника, информационные технологии, системы управления. – 2024. – № 50. – С. 146–166. DOI: 10.15593/2224-9397/2024.2.08

Please cite this article in English as:

Shmidt I.A. Database management system for working with data from bench tests of complex products. *Perm National Research Polytechnic University Bulletin. Electrotechnics, information technologies, control systems*, 2024, no. 50, pp. 146–166. DOI: 10.15593/2224-9397/2024.2.08