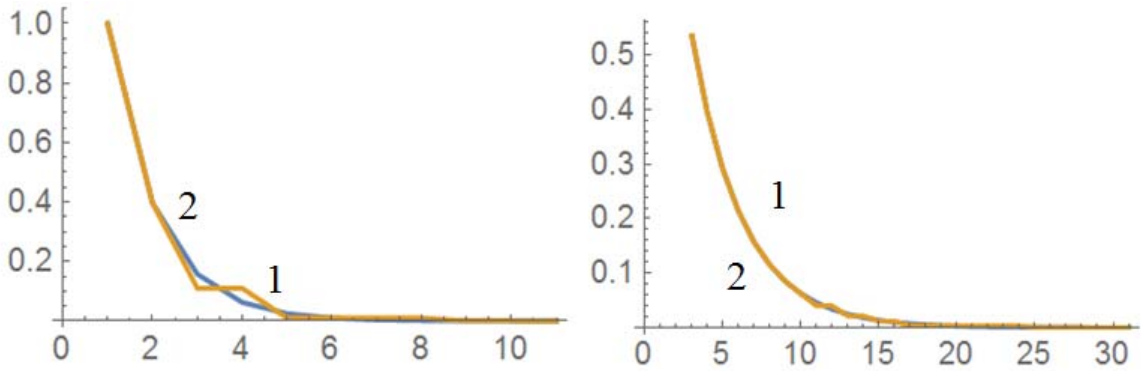

а) *б)*

Рисунок 1 – График, отражающий количество вещества в таблетке, оставшегося ко времени t при разбиении в $\Delta t = 3$ *(а)* и при разбиении в $\Delta t = 1$ *(б):*
1 – график разложения на составляющие; *2* – график исходной функции

Заметим, что на рисунке 1б относительное отклонение графиков составляет менее 1 процента. В зависимости от формы графика может быть осуществен оптимальный выбор базового вейвлета, который соответствует минимальному среднеквадратическому отклонению от заданного сигнала.

Список использованных источников

1. Астафьева Н.М. Вейвлет-анализ: основы теории и примеры применения / Н.М. Астафьева // Успехи физ. наук. – 1996. – Т. 166, №11. – С. 1145-1170.

2. Витязев В.В. Вейвлет-анализ временных рядов. – СПб.: Изд-во СПбГУ, 2001. – 58 с.

3. Воробьев В.И. Теория и практика вейвлет-преобразования / В.И.Воробьев, В.Г. Грибунин. – СПб.: ВУС, 1999. – 204 с.

4. Добеши И. Десять лекций по вейвлетам / И.Добеши. – М.: Ижевск: R\&C Dynamics, 2004. – 463 с.

5. Новиков И.Я. Основные конструкции всплесков / И.Я. Новиков, С.Б. Стечкин // Фундаментальная и прикладная математика. – 1997. – Т. 3, №4. – С. 999-1028.

6. Чуи К. Введение в вэйвлеты / К. Чуи. – М.: Мир, 2001. – 412 с.

УДК 004.514.62

СВОБОДНЫЕ СУБД ДЛЯ ХРАНЕНИЯ ВРЕМЕННЫХ РЯДОВ

Дубицкий А.В., Маркина А.А.
Брестский государственный технический университет
e-mail: alexandr.dubitsky@gmail.com

Abstract. *A review of Time Series Databases is presented, with focus on free / libre and open source software. Time-arranged list of TSDB with notes on specifics of their usage is presented as far as the generalized functionality overview and the analogy in relational database approach.*

Базы данных временных рядов (time series database, TSDB) – специализированные СУБД для хранения проиндексированных по времени данных. Причины их появления – необходимость сбора, хранения и обработки больших массивов разных метрик (системы мониторинга), а также то, что реляционные БД в системах со сложной логикой и высоким объемом транзакций для данных временных рядов – не самый практичный выбор.

Реляционные БД менее эффективны при работе с упорядоченным множеством элементов временного ряда, а другие типы ранее существовавших СУБД (например, плоские/однотабличные БД, хранящие данные в общем табличном файле) также неэффективны при большой нагрузке. Обширная сфера потенциального применения обеспечивает спрос на эффективное хранение временных рядов с возможностями обработки, характерными для реляционных БД (транзакции, математические и логические

операции над выборками данных) – и значительную роль в удовлетворении этого спроса занимают свободные проекты (табл. 1).

Для представления данных в TSDB применяются две модели. Модель «wide-table» упрощает взаимодействие с данными вне временных рядов и делает работу более привычной для разработчиков, имеющих дело с реляционными БД. Модель «narrow-table» не требует изменения структуры таблицы под новые метрики, но сложнее для восприятия человеком.

TSDB (лицензия, год)	Язык запросов	Платформа
InfluxDB (MIT, 2013)	InfluxQL, JSON via UDP	Go
Druid (Apache 2.0, 2012)	SQL, REST	Java
TimescaleDB (Apache 2.0, 2017)	SQL	C
Prometheus (Apache 2.0, 2015)	PromQL, REST	Go
KairosDB (Apache 2.0, 2013)	REST	Java

По реализации TSDB можно разделить на два типа: применение специальной схемы для хранения временных рядов в более универсальной традиционной СУБД (1) и специализированные СУБД, спроектированные с нуля (2). Заметим, что непосредственная реализация функционала TSDB в обычной реляционной БД на основе SQL возможна, если СУБД поддерживает большие двоичные объекты (BLOB) и пользовательские функции, но эффективность такой системы также будет не слишком высока.

Характерный пример TSDB первого типа – появившаяся совсем недавно TimescaleDB, которая базируется на PostgreSQL (и поэтому поддерживает SQL и все типичные функции реляционной СУБД), с оптимизациями для быстрого поиска и сложных запросов. Также TimescaleDB использует модель «wide-table», характерную скорее для реляционных СУБД, чем для TSDB. Хорошим примером TSDB второго типа является InfluxDB, первый релиз которой состоялся в 2013 году. InfluxDB упрощает анализ данных за счет собственного языка запросов InfluxQL, а также поддерживает запросы в формате JSON (большинство клиентских библиотек взаимодействуют с сервером через HTTP), и следующие типы данных: int64, float64, bool, string.

Еще один характерный пример популярной TSDB второго типа – DalmatinerDB. Механизм хранения данных этой TSDB спроектирован на базе свойств мощной и производительной файловой системы ZFS (ZettabyteFileSystem), что обеспечивает высокую эффективность хранения, но низкую переносимость. Взамен эта особенность DalmatinerDBи использованные алгоритмические решения обеспечивают масштабируемость и чрезвычайно высокую устойчивость. Язык запросов очень похож на SQL, и это снижает требования к обучению разработчиков, чего нельзя сказать о системных требованиях СУБД.

Из-за сложностей применения TSDB (привязка к конкретной СУБД, языку программирования, ОС, системные требования) разработчик может оказаться в ситуации, когда своя реализация механизма хранения временных рядов в реляционной СУБД предпочтительнее специализированного решения, хотя последнее и предоставляют лучшие функциональные возможности и оптимизацию. В таком случае реализация обычно строится в виде централизованной «таблицы фактов», хранящей данные о конкретных событиях (значения временных рядов), с уникальными составными ключами, объединяющими первичные ключи «таблиц измерений», которые содержат атрибуты событий, сохраненных в таблице фактов. Таблицысоединяются либо по схеме звезды (с уменьшением числа таблиц и ускорением запросов за счет денормализации таблиц измерений), либо по схеме снежинки (с большим числом таблиц и более полной нормализацией).

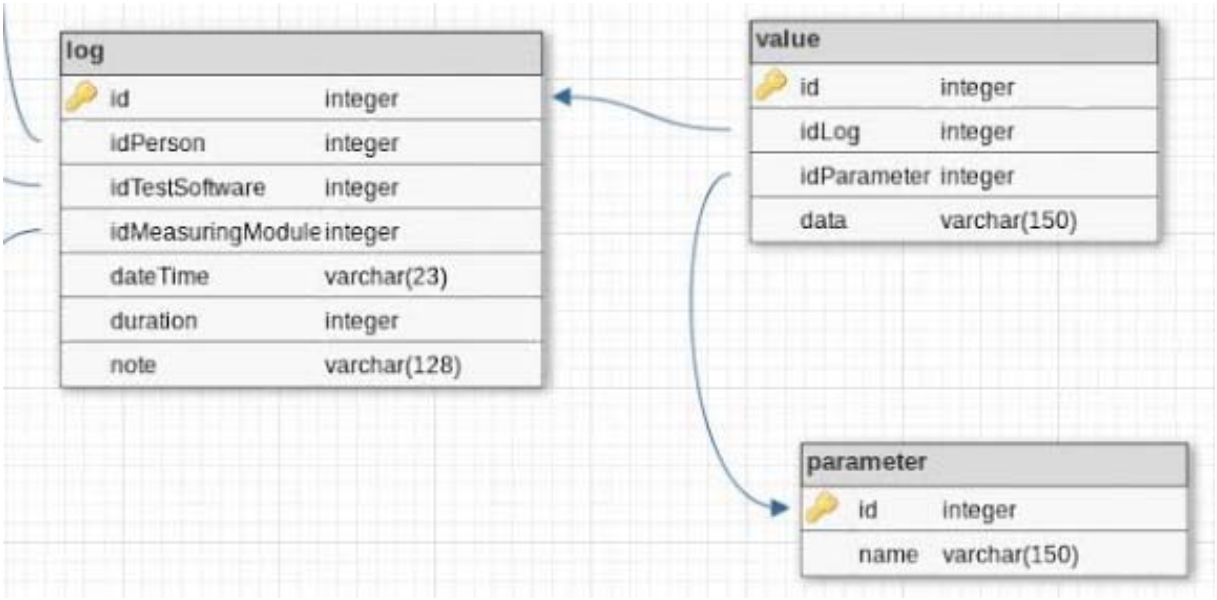

Рисунок 1 – Хранение временного ряда в реляционной СУБД (фрагмент)

В качестве примера можно рассмотреть соединение таблиц, примененное авторами в проекте UXDump для хранения результатов биометрических и иных измерений, выполняемых в ходе тестирования эффективности человеко-машинного взаимодействия. В данном проекте использована СУБД MySQL, не имеющая открытой TSDB-надстройки. В результате хранение временных рядов было реализовано следующим образом. Результаты измерений параллельно из нескольких источников заносятся в пару таблиц parameter и values (набор измеряемых параметров заранее неизвестен, что соответствует модели «narrow-table»). Таблица values, где хранятся полученные значения, связана со справочником названий измеряемых параметров parameter и с таблицей log, в которой хранится информация о сериях измерений.

Список использованных источников

1. Knowledge Base of Relational and NoSQL Database Management Systems // https://db-engines.com/en/

2. Top 10 Time Series Databases // https://blog.outlyer.com/top10-open-source-time-series-databases

3. Introduction to Time Series Database // https://www.linkedin.com/pulse/introduction-time-series-database-pinglei-guo/

4. UXDump project // https://bitbucket.org/AsyaAliset/uxdump

УДК 004.056

ИНФОРМАТИЗАЦИЯ И ОБЕСПЕЧЕНИЕ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ ТАМОЖЕННЫХ ОРГАНОВ РЕСПУБЛИКИ БЕЛАРУСЬ – ФАКТОР ПОВЫШЕНИЯ КОНКУРЕНТОСПОСОБНОСТИ

Жевлакова А.Ю., Бровка Г.М.

Белорусский национальный технический университет

Начиная с 2010 года развитие информационного общества является одним из основных факторов обеспечения конкурентоспособности и инновационного развития национальной экономики, совершенствования системы государственного управления, повышения зрелости гражданского общества.

На уровне программных документов данный приоритет закреплен в Национальной стратегии устойчивого социально-экономического развития Республики Беларусь на период до 2030 года, одобренной Президиумом Совета Министров Республики Беларусь 10 февраля 2015 г.