Кирилл В. Плаксий[1], Андрей А. Никифоров[2], Наталья Г. Милославская[3]
Национальный исследовательский ядерный университет «МИФИ»
Каширское ш., 31, Москва, 115409, Россия
[1]e-mail: kirillplaksiy@mail.ru, http://orcid.org/0000-0002-8949-6772
[2]e-mail: andreinikiforov993@gmail.com, http://orcid.org/0000-0002-2726-0000
[3]e-mail: NGMiloslavskaya@mephi.ru, http://orcid.org/0000-0002-1231-1805

ИССЛЕДОВАНИЕ ГРАФОВЫХ СУБД, ПРИГОДНЫХ ДЛЯ РАБОТЫ С БОЛЬШИМИ ДАННЫМИ ПРИ ОБНАРУЖЕНИИ ДЕЛ ПО ОТМЫВАНИЮ ДОХОДОВ, ПОЛУЧЕННЫХ ПРЕСТУПНЫМ ПУТЕМ, И ФИНАНСИРОВАНИЮ ТЕРРОРИЗМА*

DOI: http://dx.doi.org/10.26583/bit.2019.3.09

Аннотация. В статье рассматриваются популярные в настоящее время графовые системы управления базами данных (СУБД), способные работать с большими данными, с помощью которых можно реализовать хранение информации, полученной в ходе генерации преступных дел по отмыванию доходов, полученных преступным путем, и финансированию терроризма (ОД/ФТ). Цель работы заключается в выборе защищенной графовой СУБД, подходящей для работы с большими данными финансовых расследований. Для этого решаются следующие задачи: рассматриваются имеющиеся графовые СУБД, проводится их анализ и сравнение друг с другом с особым акцентом на методы защиты, используемые для обеспечения безопасности хранимых данных. Были изучены достоинства и недостатки программных продуктов, а также было проведено сравнение по ряду параметров, характеризующих защиту информации в них. Каждый критерий сравнения имеет развёрнутые комментарии, на основе которых был выбран наиболее удобный, гибкий и современный вариант СУБД для использования при поиске случаев ОД/ФТ. По полученным результатам было установлено, что графовые СУБД подходят для работы с большими данными, а также по ряду параметров была выбрана одна из рассмотренных СУБД, а именно Janus Graph.

Ключевые слова: отмывание доходов, полученных преступным путем, финансирование терроризма, ОД/ФТ, информационная безопасность, типология, большие данные, системы управления базами данных (СУБД).

Для цитирования: ПЛАКСИЙ, Кирилл В.; НИКИФОРОВ, Андрей А.; МИЛОСЛАВСКАЯ, Наталья Г. ИССЛЕДОВАНИЕ ГРАФОВЫХ СУБД, ПРИГОДНЫХ ДЛЯ РАБОТЫ С БОЛЬШИМИ ДАННЫМИ ПРИ ОБНАРУЖЕНИИ ДЕЛ ПО ОТМЫВАНИЮ ДОХОДОВ, ПОЛУЧЕННЫХ ПРЕСТУПНЫМ ПУТЕМ, И ФИНАНСИРОВАНИЮ ТЕРРОРИЗМА. Безопасность информационных технологий, [S.l.], v. 26, n. 3, p. 103-116, 2019. ISSN 2074-7136. Доступно на: <https://bit.mephi.ru/index.php/bit/article/view/1222>. Дата доступа: 17 sep. 2019. doi:http://dx.doi.org/10.26583/bit.2019.3.09.

***Благодарности.** Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 18-07-00088.*

Kirill V. Plaksiy[1], Andrey A. Nikiforov[2], Natalia G. Miloslavskaya[3]
National Research Nuclear University MEPhI,
Kashirskoe shosse, 31 Moscow, 115409, Russia
[1]e-mail: kirillplaksiy@mail.ru, http://orcid.org/0000-0002-8949-6772
[2]e-mail: andreinikiforov993@gmail.com, http://orcid.org/0000-0002-2726-0000
[3]e-mail: NGMiloslavskaya@mephi.ru, http://orcid.org/0000-0002-1231-1805

Investigation of graph databases suitable for work with big data while detecting money laundering and terrorism financing cases*

DOI: http://dx.doi.org/10.26583/bit.2019.3.09

Abstract. This paper discusses the currently popular graph database management systems (DBMSs) working with Big Data and can be used to store information obtained while dealing with money laundering and terrorist financing criminal (ML/FT) cases. The aim of this study is to choose a secure graph DBMS suitable for working with Big Data for such financial investigations. The authors consider the existing graph DBMSs, analyze and compare them with each other with special emphasis on the information security protection methods of stored data. The advantages and disadvantages of software products are studied and a comparison with the help of selected parameters characterizing system's ability to keep information secure is made. The results of the comparison are followed by detailed comments. On its basis the most convenient, flexible and up-to-date DBMS was chosen for usage while searching ML/FT cases. It was found that graph DBMSs are suitable for Big data tasks and as a final result JanusGraph was selected as a foreground DBMS in this project according to selected parameters.
Keywords: money laundering, terrorism financing, ML/FT, information security, typology, Big Data, Database Management System (DBMS).
For citation: PLAKSIY, Kirill V.; NIKIFOROV, Andrey A.; MILOSLAVSKAYA, Natalia G. Investigation of graph databases suitable for work with big data while detecting money laundering and terrorism financing cases. IT Security (Russia), [S.l.], v. 26, n. 3, p. 103-116, 2019. ISSN 2074-7136. Available at: <https://bit.mephi.ru/index.php/bit/article/view/1222>. Date accessed: 17 sep. 2019. doi:http://dx.doi.org/10.26583/bit.2019.3.09.

***Acknowledgement.** The research was carried out under the financial support of the RFBR in the framework of scientific project No. 18-07-00088.*

Введение

С каждым годом количество производимых человеком и техникой данных резко возрастает. Сначала таблицы и графы насчитывали в себе сотни или тысячи значений и вершин и для их хранения использовались обычные базы данных (БД). С развитием технологий и возрастанием объёмов получаемых сведений появился термин «Большие данные» (англ. Big Data), что дало толчок развитию новых средств сбора, обработки и хранения информации. Это подвело учёных к необходимости пересмотра существующих хранилищ с целью создания современных систем управления БД, способных отвечать актуальным запросам бизнеса и разносторонних исследований. При этом от простых числовых значений разработчики пришли к увеличению множества типов данных, которые могут храниться в таких БД. Так, из-за нехватки памяти для уменьшения избыточности данных и сохранения их целостности появилась реляционная СУБД [1], основанная на теории нормализации.

Интерес научного сообщества к подобным технологиям огромен. В последнее десятилетие было разработано множество инструментов для обработки больших объемов информации, условно разделяемых на структурированные (длина и формат данных четко определены, обычно хранятся в реляционных СУБД), неструктурированные (данные без определенного формата, сохраняются в том виде, в котором они были собраны) и полуструктурированные данные (данные, которые не ограничиваются определенными полями, но имеют маркеры для разделения информации, например, как в стандарте JSON (JavaScript Object Notation) [2]).

Различные типы данных могут храниться в разных форматах. Использование универсального хранилища со множеством форматов требуется в случаях, когда речь идет о больших объемах крайне изменчивой информации, например, в социальных сетях.

Способность графовых СУБД связывать информацию определяет их применимость для задач информационной безопасности (ИБ), например, для обнаружения мошенничества, управления процессом аутентификации, управления угрозами ИБ и т.д. Специалист с развитым умением понимать структуру и анализировать содержание

подобных БД получает большие возможности для исследования, так как эти СУБД наглядно демонстрируют ошибки в защите и взаимосвязи объектов, которые могут в дальнейшем повлиять на другие смежные системы в организации.

Графовые СУБД активно используются в финансовых организациях как для поддержки и контроля бизнес-процессов, так и для обеспечения их ИБ [3]. Развитая инфраструктура вкупе с созданными под конкретные задачи хранилищами повышает успех обнаружения инцидентов ИБ и уменьшает время успешного реагирования на них. Но злоумышленники могут применить координально новый подход к преступлению или модернизируют его схему. Тогда необходимо заранее иметь комплекс превентивных мер и средств защиты от него.

Финансовые преступления, а точнее, результаты их анализа и каталогизации можно отнести к большим данным, так как они характеризуются их тремя базовыми свойствами: volume – большим объёмом данных, velocity – обработкой информации с большой скоростью, variety – многообразием и неструктурированностью данных. При рассмотрении больших схем преступлений с сотней участников (физлиц, финансовых учреждений, индивидуальных предпринимателей) у всех будут свои идентификационные данные, различные логические связи, множество проведенных денежных операций и т.п. При этом возможны модифицированные и комбинированные схемы преступлений.

Данная статья продолжает работы, посвященные использованию графов в расследованиях финансовых преступлений, а именно генерации вариантов типологий (наиболее распространенных схем) для последующего их использования в поиске схем отмывания денег, полученных преступным путем, и финансирования терроризма (ОД/ФТ) [4]. Ее актуальность обусловлена необходимостью обоснованного выбора защищенных графовых СУБД, подходящих для этих целей и работы с большими данными финансовых расследований. Для этого решаются следующие задачи: рассматриваются имеющиеся графовые СУБД, проводится их анализ и сравнение друг с другом с особым акцентом на методы защиты, используемые для обеспечения безопасности хранимых данных. Для достижения поставленной цели статья структурирована следующим образом: сначала кратко проанализированы работы в данной области, далее выделены распространенные методы защиты данных в СУБД, представлены наиболее известные графовые хранилища, приведено сравнение и выбор подходящей СУБД.

1. Работы в данной области

В ходе исследования были рассмотрены различные графовые хранилища данных, которые могут быть применены к делам ОД/ФТ. Графы как средство визуализации используются для работы с данными [5], что упрощает восприятие человеком информации и уменьшает её объемы за счет различных подходов, учитывающих специфику решаемой задачи. Для выполнения всевозможных операций над данными графы начали применяться и в программировании, что было описано еще в начале XXI века [6].

В сфере противодействия ОД/ФТ (ПОД/ФТ) графы также используются, позволяя избавиться от незначимых в рамках решаемых задач данных и оставляя только необходимую для анализа информацию [7]. Поскольку данные о транзакциях и различных операциях давно превзошли пределы обычных данных, то для ПОД/ФТ стали применяться инструменты больших данных, например, система SAS Anti-Money Laundering (SAS AML), специально созданная для банковских систем, отслеживания операций банка, консолидации информации и выявления подозрительных действий [8].

В графовых СУБД информация хранится в виде узлов или объектов, а отношения между ними позволяют получать дополнительную информацию, имеющую большую ценность. Несмотря на то, что эта технология является относительно новой, наиболее значимые результаты получаются с 2013 года, и уже существуют СУБД, которые работают в реальных системах. Для хранения новых данных и работы с ними используются, например, платформы Neo4j [9], DataStax Enterprise Graph [10], AllegroGraph, ArangoDB и другие.

Про графовые СУБД доступно много публикаций разных авторов. Например, в [11] обосновывается актуальность и необходимость использования графовых БД, а также сравниваются популярные в 2018 году решения. В [12] описаны цели и способы проектирования и реализации графовых баз, а также показаны не совсем стандартные случаи применения. В [13] приводится небольшой раздел по графовым базам и их сравнение между собой. В [14] показаны возможности построения новой платформы для моделирования с целью отображения уязвимостей в сетях. Эти и другие публикации были положены в основу данного исследования.

2. Распространённые методы защиты данных в СУБД

С ростом потребностей в операциях с информацией возрастала необходимость в средствах обеспечения ИБ этой информации в БД. Средства защиты в различных СУБД несколько отличаются, однако общим является многоуровневость защиты – чем больше барьеров-уровней, тем сложнее будет их преодолеть злоумышленнику. На нижних уровнях находятся стандартные способы защиты: пароли, шифрование данных, разграничение прав доступа к объектам БД, контрольные следы выполняемых операций, резервное копирование.

Перечисленные способы являются частью более общей классификации уровней безопасности. Согласно «Критериям определения безопасности компьютерных систем» [15] определяются четыре класса безопасности (Security Classes): D, C, B и A. Класс D обеспечивает минимальную защиту (Minimal Protection), класс C – дискреционную (Discretionary), класс B – мандатную (Mandatory), а класс A – верифицируемую (Verified).

Дискреционное управление доступом осуществляется по усмотрению владельца данных и делится на 2 подкласса – C1 и C2, где подкласс C1 является менее безопасным, чем подкласс C2. Требованием класса C1 является разделение данных и пользователя, помимо взаимного доступа к данным возможно их раздельное использование пользователями. Класс C2 предусматривает дополнительный учет на основе входа в систему, аудита и изоляции ресурсов. Дискреционное управление доступом поддерживается многими СУБД и базируется на идентификации пользователей, объектах БД (таблицах, представлениях, доменах, определенных пользователем наборе символов, хранимых процедурах и т.д.) и привилегиях – наборе действий над тем или иным объектом.

При мандатном управлении доступом объектам данных присваиваются определенные классификационные уровни, образующие строгий иерархический порядок (например, «секретно», «совершенно секретно», «для служебного пользования» и т.д.), а каждый пользователь имеет соответствующий уровень допуска. Защита класса B делится на три подкласса – B1, B2 и B3, где подкласс B1 наименее безопасен, а B3 является наиболее безопасным подклассом. В соответствии с требованиями класса B1 каждый объект данных содержит отметку о его уровне классификации, а также неформальное сообщение о действующей политике безопасности. Согласно требованиям класса B2 дополнительно требуется формальное утверждение о действующей политике

безопасности. Кроме того, необходимо решить вопрос о защищённых каналах передачи информации. Наконец, класс B3 помимо прочего требует поддержки аудита, восстановления данных и назначения администратора режима безопасности.

Верифицируемая защита класса A является наиболее безопасной, но требует математического доказательства соответствия метода обеспечения безопасности заданной политике.

Шифрование данных. Существуют два режима работы с зашифрованными БД. Первый заключается в расшифровании необходимого файла или части файла на внешнем носителе. После выполнения необходимых действий с открытой информацией она вновь зашифровывается на внешнем запоминающем устройстве. Независимое последовательное функционирование средств шифрования и СУБД является несомненным достоинством такого режима. Однако в результате сбоя или отказа часть БД может остаться записанной в незашифрованном виде. Расшифрование также может проводиться в оперативной памяти непосредственно перед выполнением необходимых действий с данными. Процедуры шифрования часто встроены в СУБД. Необходимо отметить, что несмотря на достаточно высокий уровень защиты от несанкционированного доступа, снижается уровень производительности СУБД в связи с ее усложнением.

Защита полей. Поскольку в большинстве случаев изменение информации в СУБД является следствием человеческих действий, целесообразно применять защиту полей и записей в таблицах и формах. Защита данных в полях таблиц подразумевает следующие уровни прав доступа: полный запрет доступа, только чтение и разрешение всех операций (просмотр, ввод, удаление и изменение). Некоторые поля готовых таблиц могут быть скрыты для ряда пользователей. Для экранных форм готовых приложений обычно запрещают вызов конструктора, чтобы конечный пользователь случайно не изменил приложение. В самих экранных формах отдельные элементы также могут быть защищены.

Контрольный след выполняемых операций позволяет регистрировать детальные сведения обо всех операциях пользователей с БД. Такая сохраненная информация играет существенную роль в обнаружении несанкционированного вмешательства в БД, выявлении уязвимостей в системе защиты и устранении каких-либо внесенных искажений данных.

Резервное копирование позволяет восстанавливать данные на случай аппаратных или программных сбоев.

Также существуют средства защиты, присущие какой-то одной конкретной БД, что делает её функционал уникальным, например, как это сделано в тройных атрибутах в семантической графовой БД AllegroGraph [16]. Это позволяет сделать данные прозрачными для пользователей на основе ролей. Они обеспечивают возможность связывания графовых СУБД с защитой доступа к данным. Пока тройка атрибутов задействована в установленной уже связи, их не получится изменить без изменения самой связи.

3. Современные графовые хранилища

В настоящее время существуют десятки графовых СУБД, из которых наиболее известными сейчас являются AllegroGraph (мультитиповая), ArangoDB, FlockDB, Giraph, HyperGraphDB, InfiniteGraph, InfoGrid, Neo4j, OrientDB, SparkSee (ранее DEX), Sqrrl, Titan, Datomic, JanusGraph. По данным [17] на август 2019 года среди именно графовых СУБД лидирующие позиции занимает Neo4j, 6 место – JanusGraph, 8 и 9 делят Dgraph и Giraph, соответственно, на 11 располагается TigerGraph, 13 – AllegroGraph, на 17 – InfiniteGraph, 19 место отдано FlockDB, 21 – InfoGrid, 23 – HyperGraphDB, с 25 по 28 занимают Sparksee,

TinkerGraph, GraphBase, HGraphDB, а общий рейтинг на 30–32 местах замыкают Memgraph, DataChemist и FlureeDB.

Neo4j [18] представляет из себя свободно распространяемую графовую СУБД с открытым исходным кодом. Разработчиком является одноимённая компания, которая начала работу над проектом в 2003 г., а в 2007 году представила первую готовую версию. На сегодняшний день последней версией продукта является 3.5.4 от апреля 2019 года Основными языками реализации являются Java и Scala.

JanusGraph [19] – масштабируемая графовая БД, оптимизированная для их хранения и обработки, причем графы могут содержать сотни миллионов вершин и ребер, распределенных по кластеру из нескольких машин. Реализовано удобное взаимодействие с другими СУБД (Apache Cassandra, Apache HBase, Google Cloud Bigtable, Oracle BerkeleyDB). Масштабируемость зависит от технологий, которые используются вместе с СУБД. Например, используя Apache Cassandra в качестве хранилища, масштабируемость до нескольких центров обработки данных предоставляется «из коробки».

Dgraph [20] – свободная, масштабируемая, распределённая графоориентированная СУБД. Она развивается с учетом обеспечения минимальных задержек выполнения запросов, что позволяет использовать её для обработки информации в режиме реального времени. Архитектура поддерживает создание распределённых конфигураций из нескольких экземпляров Dgraph, давая возможность масштабировать хранилища путём добавления дополнительных узлов при росте нагрузки или увеличении размера данных.

TigerGraph [21] представляет собой одну из графических СУБД нового типа: это первая система, способная выполнять анализ данных в режиме реального времени в масштабе веб-сети. Дизайн Native Parallel Graph (NPG) ориентирован как на хранение, так и на вычисления, поддерживает обновление графиков в реальном времени и предлагает встроенные параллельные вычисления. Язык запросов SQL-подобных графов (GSQL) обеспечивает специальное исследование и интерактивный анализ больших данных. Благодаря возможностям GSQL и скорости NPG пользователь может выполнять аналитику Deep Link: обнаруживать соединения, слишком громоздкие в плане обработки информации.

AllegroGraph [22] – высокопроизводительная графическая (мульти) СУБД с закрытым исходным кодом. AllegroGraph совмещает эффективное использование оперативной памяти с использованием дисковых хранилищ. Поддерживает языки запросов к данным SPARQL, RDFS++ и Prolog и автоматически использует те из них, которые совместимы с приложениями пользователя. В настоящее время используется в проектах с открытым исходным кодом, в коммерческих проектах и проектах Министерства обороны США. Является компонентой хранения данных в проекте TwitLogic, реализующем концепцию Семантической паутины в системе обработки данных социальной сети Twitter.

Все приведенные СУБД имеют большой потенциал для работы с большими данными, обладают нужным функционалом и наполнением. Ниже представлена сравнительная таблица (табл. 1), в которой акцент был сделан на методах защиты, реализованных в выбранных СУБД. Критерии для сравнения были выбраны с учетом того, что СУБД должна иметь механизмы защиты данных, которые не затрудняли бы работу с самой СУБД в условиях проводимого исследования. Чем более гибко реализована возможность настройки защиты, тем лучше.

Таблица 1. Сравнение популярных графовых СУБД

	Neo4j	**JanusGraph**	**Dgraph**	**TigerGraph**	**AllegroGraph**
Распространение	Открытый исходный код	Открытый исходный код	Открытый исходный код	Коммерческая лицензия	Коммерческая лицензия
Языки реализации	Java, Scala	Java	Go	C++	Java, Python, Common Lisp
Серверные операционные системы	Linux, OS X, Solaris, Windows Может использоваться и без сервера в качестве встроенной базы данных Java	Linux, OS X, Unix, Windows	Linux, OS X, Windows	Linux	Linux, OS X, Windows
Контроль доступа	Присутствует	Осуществляется через Rexter Graph Server	Нет (запланировано на будущие версии)	Ролевая система управления доступом	Ролевая система управления доступом
Аутентификация	Подключаемая аутентификация с поддерживаемыми стандартами (LDAP, Active Directory, Kerberos)	Базовая и токен-аутентификация	Нет (запланировано на будущие версии)	Аутентификация GSQL и REST ++, токены	Базовая аутентификация с настройкой дополнительных фильтров
Шифрование	Целенаправленного внутреннего шифрования нет, поддержка стороннего шифрования	Целенаправленного внутреннего шифрования нет, поддержка стороннего шифрования	Шифрование данных в состоянии покоя HDFS	Запатентованная схема шифрования + поддержка стандартных методов шифрования	FIPS 140-2 шифрование для передачи данных
Целостность данных	Использование ограничений	Контроль целостности данных при загрузке	Использование двойных баз	Пока не реализовано	Запись всех процессов в журналы
Резервные копии	Да, как для одной машины, так и для кластеров	Собственные копии, а также поддержка сторонних средств	Копии работающих кластеров	Интегрированный инструмент для резервного копирования и восстановления данных и словаря данных одного узла	Поддержка создания копий онлайн

Помимо представленной таблицы 1 для выбора хранилища информации в исследовании интерес представляют те преимущества и недостатки (табл. 2), которые могут быть важны для хранения сгенерированных графовых данных типологий финансовых преступлений.

Таблица 2. Достоинства и недостатки популярных графовых СУБД

Название	Достоинства	Недостатки
Neo4j	• Простой язык запросов и может предоставлять наглядный результат, что удобно в решении аналитических задач; • гибкая структура данных, что позволяет вносить изменения в случае изменения требований; • можно сразу создавать модели, которые приближены к реальным условиям без низкоуровневых деталей; • высокая производительность при использовании специфических моделей данных и легкость работы с ними	• Места на диске уходит больше по сравнению с реляционными СУБД; • простые запросы до определенного уровня имеют более низкую эффективность выполнения (производительность) нежели в реляционных базах
JanusGraph	• Поддержка очень больших графов, которые масштабируются в зависимости от количества машин в кластере; • поддержка большого числа параллельных транзакций. Транзакционная емкость масштабируется в зависимости от количества машин в кластере и отвечает на сложные запросы обхода на огромных графах за миллисекунды; • поддержка глобальной аналитики графов и пакетной обработки графов через платформу Hadoop; • поддержка полнотекстового поиска для вершин и ребер на очень больших графах; • многочисленные конфигурации на уровне графов для улучшения производительности; • оптимизированное представление диска, что позволяет эффективно использовать хранилище и скорость доступа; • отдельные преимущества при использовании СУБД вместе с конкретными инструментами, например, с Cassandra или с HBase	• Трудно сделать ручное управление транзакциями; • понижение производительности при добавлении вершин при наличии большого количества уже существующих
Dgraph	• Разбиение данных горизонтально на сотни серверов, что предназначено для минимизации количества обращений к диску и сетевых вызовов; • высокая скорость работы. Dgraph построена как поисковая система – запросы разбиваются на подзапросы, которые запускаются одновременно для достижения низкой задержки и высокой пропускной способности; • соответствие требованиям ACID (Atomicity, Consistency, Isolation, Durability) – нет необходимости беспокоиться о целостности данных; • автоматический запуск синхронной репликации, поэтому потеря жесткого диска или сервера не влияет на сервисы; • равномерная сбалансированность данных по серверам путем автоматического размещения и улучшения использования ресурсов для высокой производительности; • пользовательский интерфейс для легкого просмотра и управления данными; • эффективное использование аппаратных средств хранения. Внутреннее хранилище ключей Dgraph, Badger, предназначено для уменьшения использования ОЗУ. Оптимизация на SSD (solid-state drive) повышает производительность.	• Сложность выполнения больших аналитических запросов; • большое значение имеет задание связей между узлами графа, что влияет на эффективность поиска; • кластер состоит из разных компонентов (zero, server и ratel), и каждый компонент предназначен для своей цели, что без должного внимания и определения функций компонентов снижается эффективность работы

Название	Достоинства	Недостатки
TigerGraph	• Быстрая загрузка данных для быстрого построения графов; • ускоренное выполнение алгоритмов параллельных графов; • возможность унифицировать аналитику в реальном времени с крупномасштабной автономной обработкой данных; • возможность масштабирования для распределенных приложений; • возможность прохождения огромного количества вершин/ребер в секунду и загрузки от 50 до 150 ГБ данных в час (на машину)	• При наличии низкоуровневых ошибок возможна каскадная реакция, что затрудняет поиск неправильных фрагментов графа или зависимости; • выбор систем на серверах и используемые языки жёстко ограничены небольшим набором
AllegroGraph	• Можно смешивать геопространственную, временную, социальную сетевую аналитику и анализ, все в одном запросе (SPARQL или Prolog); • трехуровневая безопасность с фильтрами; • резервное копирование в онлайн-хранилище, восстановление на момент времени, репликация	• Занятые тройные атрибуты могут привести к усложнению логики запросов из-за их текущего задействования в работе СУБД

4. Сравнение СУБД и обсуждение результатов

На основе данных из открытых источников, выявленных выше особенностях СУБД и с учётом приоритетов проводимого исследования осуществлялся выбор графовой СУБД. Для удобства сравнения СУБД по обеспечению ИБ ниже по каждой из строк таблицы даны комментарии и выделены приоритетные решения.

Аутентификация и контроль доступа.

Ядро модели безопасности Neo4j сосредоточено вокруг ряда предопределенных ролей доступа к данным на глобальном уровне. Каждая роль включает в себя набор действий, разрешенных для графа данных. Neo4j предоставляет собственного поставщика аутентификации, который локально хранит информацию о пользователях и ролях на диске. Другой способ управления аутентификацией и авторизацией – через внешнее ПО, такое как Active Directory или OpenLDAP, доступ к которому осуществляется через встроенный соединитель LDAP. В СУБД предоставляется опция плагина для создания пользовательских интеграций. В дополнение к LDAP (Lightweight Directory Access Protocol), собственным и пользовательским решениям, для аутентификации и единого входа Neo4j поддерживает Kerberos. Поддержка Kerberos обеспечивается с помощью дополнения Neo4j Kerberos.

JanusGraph поддерживает соединения, которые используют HTTP-запросы или через WebSockets. Все соединения выполняются по HTTPS (HyperText Transfer Protocol Secure) и требуют аутентификации. HTTP-запросы поддерживают базовую и токен-аутентификацию. Для чего-либо, кроме разового вызова, использование WebSockets или аутентификации по токенам приводит к повышению производительности. Обычная аутентификация для СУБД, когда имя пользователя и пароль отправляются вместе с запросом, – это дорогостоящий по ресурсам процесс.

При первой инсталляции TigerGraph аутентификация пользователя отключена. В процессе установки создается суперпользователь gsql, у которого есть профиль с именем и паролем. Пока пароль пользователя tigergraph равен tigergraph, аутентификация gsql остается отключенной. Это разработано для удобства пользователя в однопользовательских конфигурациях или установках типа демонстрационных и обучающих примеров, не требующих защиты.

В AllegroGraph реализована ролевая система управления доступом с поддержкой различных фильтров безопасности, что позволяет установить гибкие ограничения и минимизировать потери производительности.

Рассмотрев аутентификацию и контроль доступа в данных СУБД, а также принимая во внимание приоритетность выполнения запросов перед их скоростью, как оптимальное решение выбираем JanusGraph.

Шифрование.

Хотя в настоящее время Neo4j не занимается явным шифрованием данных, для сценариев, где требуется дополнительная безопасность, распространены два подхода: шифрование файловой системы, в которой находится БД, и шифрование самих данных из приложения. Шифрование файловой системы – это простой, полезный шаг, который приводит к усилению защиты данных на диске. Однако одного такого шифрования недостаточно для полной защиты данных. Это связано с тем, что Neo4j использует архитектуру на основе REST (Representational State Transfer), которая отвечает на операторы Cypher, отправленные в виде вызовов веб-службы, ответы на эти вызовы (то есть данные) передаются по Сети в виде открытого текста. Хотя следующим логическим шагом является использование HTTPS для шифрования сетевого взаимодействия, некоторые приложения требуют дополнительной защиты, такой как ограничение доступа к данным только тем, кто авторизован для работы с ними. С этого начинается шифрование на уровне приложений – процесс, при котором приложение динамически изменяет данные во время выполнения, выполняя шифрование и расшифрование данных до и после записи или чтения данных из БД. Многие широко признанные отраслевые стандарты безопасности, например, такие как HIPPA (Health Insurance Portability and Accountability Act) и FERPA (Family Educational Rights and Privacy Act) для областей здравоохранения и образования, могут быть реализованы с помощью защиты на уровне приложений. В случае приложений на основе Java библиотека Neo4j Object Graph Mapping (OGM) может использоваться для реализации безопасности на уровне приложений. Поскольку Neo4j может сохранять данные в формате, отличном от формата коренной модели (например, сохранение даты в виде Long или String), OGM предлагает концепцию конвертации атрибутов. Создание пользовательской реализации подобных преобразований является хорошей отправной точкой для шифрования на уровне приложений.

Достоинство – простой в применении подход приводит к детальному шифрованию, которое обеспечивает безопасность данных как на диске, так и во время передачи по Сети на сервер Neo4j и от него. Конкретный используемый процесс шифрования данных может быть полностью адаптирован в соответствии с конкретными потребностями.

Недостаток – дополнительная защита не улучшает производительности системы. Сам процесс шифрования данных влечет за собой вычислительные затраты с точки зрения памяти и вычислительной мощности. Тот факт, что данные передаются в своем зашифрованном формате (который обычно намного больше, чем его открытый текст), отрицательно скажется на использовании Сети. По самой природе зашифрованные данные становятся более трудными для работы за пределами приложения, и функции БД, такие как индексы, поиск и случайные запросы Cypher, становятся нежизнеспособными. Наконец, существующие данные необходимо преобразовать в желаемый зашифрованный формат, чтобы они совпадали с требуемым форматом, поэтому следует позаботиться о том, чтобы приложение могло успешно работать как с зашифрованными, так и с незашифрованными значениями.

У JanusGraph шифрование отдано на реализацию тому продукту, с которым осуществляется взаимодействие, а таких достаточно много. Можно найти компромисс между оперативностью выполнения запросов и шифрованием.

В Dgraph предусмотрена функция шифрования данных в состоянии покоя HDFS (Hadoop Distributed File System), которая, если она включена, позволяет хранить данные в зашифрованных каталогах HDFS, называемых зонами шифрования. Все файлы в зоне шифрования прозрачно шифруются и расшифровываются на стороне клиента. Следовательно, расшифрованные данные никогда не хранятся в HDFS.

TigerGraph использует запатентованную схему шифрования, которая сжимает и скрывает данные, если пользователь не знает схему шифрования/расшифрования. Кроме того, система TigerGraph поддерживает интеграцию со стандартными методами шифрования данных при хранении на диске. Шифрование данных в состоянии покоя может применяться по-разному на усмотрение пользователя. Также это осуществимо в режиме ядра. Для запуска в режиме ядра требуется разрешение суперпользователя. Шифрование файловой системы использует передовые алгоритмы шифрования, например AES. Шифрование обычно связано с процессором, а не с вводом/выводом. Если загрузка процессора ниже 100%, тесты TigerGraph не показывают существенного снижения производительности.

Поскольку AllegroGraph широко используется в силовых структурах США, в ней интегрировано шифрование с использованием FIPS 140-2. Этот стандарт не распространен в России, к тому же в последнее время он подвергается критике. Целесообразно будет дождаться окончательной оценки экспертов по этому вопросу.

Проанализировав возможности СУБД по шифрованию и оценив их потенциал, сделали вывод, что приоритетными вариантами по данному параметру в условиях большого проекта являются JanusGraph и Neo4j.

Целостность данных.

Neo4j помогает обеспечить целостность данных с использованием ограничений, применяемых к узлам или отношениям. Могут быть созданы уникальные ограничения свойств узлов, а также ограничения существования свойств узлов и свойств отношений. Также могут быть реализованы ключи узлов, которые гарантируют как существование, так и уникальность. Ограничения уникальных свойств гарантируют, что значения свойств являются уникальными для всех узлов с определенной меткой. Уникальные ограничения не означают, что все узлы должны иметь уникальное значение для свойств – узлы без свойства не подчиняются этому правилу. Ограничения существования свойства гарантируют, что свойство существует для всех узлов с определенной меткой или для всех отношений с определенным типом. Все запросы, которые пытаются создать новые узлы или отношения без свойства, или запросы, которые пытаются удалить обязательное свойство, теперь не будут выполнены. Ключи узла гарантируют, что для данной метки и набора свойств все свойства существуют на всех узлах с этой меткой и сочетание значений свойств является уникальным.

Как и в предыдущем случае, в JanusGraph целостность данных контролируется дополнительными продуктами. В самой же СУБД осуществляется контроль целостности данных при загрузке.

В Dgraph хранилища данных графа представляются как слой графа над какой-либо другой БД SQL/NoSQL для управления данными. Эта другая БД отвечает за резервное копирование, моментальные снимки, сбои сервера и целостность данных.

AllegroGraph поддерживает целостность данных с помощью журналирования на момент их появления в общем реестре, а также в ходе их изменения. Целостность во

время выполнения процессов обеспечивается сторонними модулями, что позволяет иметь разные журналы и различные точки зрения в случае возникновения инцидентов.

По параметру обеспечения целостности в рамках проекта интересными считаются JanusGraph и AllegroGraph ввиду гибкости настройки контроля целостности и взаимодействия с другими программными продуктами.

Резервное копирование.

Почти все СУБД имеют похожие по принципу действия инструменты для создания резервных копий. Возможны копии как одной машины, так и кластера. У некоторых осуществимо копирование в режиме онлайн. Отдельного упоминания стоит инструмент, разработанный целенаправленно для одной из рассмотренных баз.

GBAR (Graph Backup and Restore) – это интегрированный инструмент для резервного копирования и восстановления данных и словаря данных (схемы, загрузки заданий и запросов) одного узла TigerGraph. В режиме резервного копирования он упаковывает данные TigerGraph и информацию о конфигурации в один файл на диск или в удаленную корзину AWS S3. Несколько файлов резервных копий могут быть заархивированы. Позже можно использовать режим восстановления для отката системы на любую точку резервного копирования. Этот инструмент также можно легко интегрировать с Linux Cron для выполнения периодических заданий резервного копирования.

По параметру резервного копирования нельзя однозначно выделить лидера, так как данная функция является одной из жизненно важных для СУБД. Поэтому в данном разделе выбор любой из СУБД будет хорошим вариантом.

В результате проведенного исследования для выполнения поставленной задачи хранения данных финансовых преступлений (сгенерированных на основе типологий объектов и логических связей между ними) была выбрана JanusGraph из-за её преимуществ перед другими СУБД. Среди ее достоинств были выделены масштабируемая обработка графовых данных и выполнение аналитических запросов в реальном времени, которая пригодится при сравнении реальных операций со сгенерированными, обработка графов через платформу Hadoop (данная платформа является самой популярной для работы с большими данными), полнотекстовый поиск для вершин и ребер на очень больших графах, взаимодействие с различными популярными инструментами больших данных и работа под основными операционными системами и с самыми популярными языками программирования.

Заключение

В работе проведено исследование некоторых наиболее популярных графовых СУБД с целью выбора реализации, наиболее подходящей для решения задач ПОД/ФТ. По полученным результатам было установлено, что графовые СУБД подходят для задач работы с большими данными, а также была выбрана одна из рассмотренных СУБД, а именно Janus Graph по выше перечисленным параметрам.

В будущем планируется заполнение выбранной СУБД сгенерированными данными финансовых преступлений, их проверка на реалистичность, анализ прошедших отбор для повышения эффективности поиска финансовых нарушений, а также выявление новых преступных схем ОД/ФТ.

СПИСОК ЛИТЕРАТУРЫ:

1. National Research Council et al. Funding a revolution: Government support for computing research. National Academies Press, 1999.
2. Understanding JSON Schema. URL: https://json-schema.org/understanding-json-schema/reference/type.html (дата обращения: 12.08.2019).
3. Eifrem E. Why Graph Database Could Be Key to Addressing Financial Services Challenges. URL: https://financialit.net/blog/financial-services/why-graph-database-could-be-key-addressing-financial-services-challenges (дата обращения: 15.08.2019).
4. Plaksiy K., Nikiforov A., Miloslavskaya N. Applying Big Data Technologies to Detect Cases of Money Laundering and Counter Financing of Terrorism. Proceedings of 2018 6th International Conference on Future Internet of Things and Cloud (FiCloud2018). Barcelona (Spain), 6-8 August 2018. P. 70–77. DOI 10.1109/W-FiCloud.2018.00017.
5. Харари Ф. Теория графов. – М.: Мир, 1973. – 296 с.
6. Касьянов В.Н. Визуализация информации на основе графовых моделей. Научная визуализация. В.Н. Касьянов, Е.В. Касьянова. – М.: Наука, 2014, Т.6. № 1. С. 31–50.
7. Drezewski R., Sepielak J., Filipkowski W. The Application of Social Network Analysis Algorithms in a System Supporting Money Laundering Detection. In Digital Investigation June 2012 9(1). P. 8–21. Elsevier Ltd.
8. SAS Anti-Money Laundering (SAS AML). URL:http://www.tadviser.ru/index.php/%D0%9F%D1%80%D0%BE%D0%B4%D1%83%D0%BA%D1%82:SAS_Anti-Money_Laundering_(SAS_AML) (дата обращения: 14.08.2019).
9. Neo4j – Platform for connected data. URL: https://neo4j.com/ (дата обращения: 02.08.2019).
10. DataStax Enterprise Graph Super-powering your data relationships. URL: https://www.datastax.com/products/datastax-enterprise-graph/ (дата обращения: 03.08.2019).
11. Fernandes D., Bernardino J. Graph Databases Comparison: AllegroGraph, ArangoDB, InfiniteGraph, Neo4J, and OrientDB. DATA. 2018. P. 373–380.
12. Robinson I., Webber J., Eifrem E. Graph databases. O'Reilly Media. 2013.
13. Tabacchi M. E. et al. Designing Cognitive Cities. Designing Cognitive Cities. Springer, Cham, 2019. P. 3–27.
14. Noel S. et al. Big-data architecture for cyber attack graphs representing security relationships in nosql graph databases. 2015.
15. Qiu L. et al. Trusted computer system evaluation criteria. National Computer Security Center. 1985.
16. AllegroGraph 6.6.0 Triple Attributes. URL: https://franz.com/agraph/support/documentation/current/triple-attributes.html (дата обращения: 02.08.2019).
17. DB-Engines Ranking of Graph DBMS. URL: https://db-engines.com/en/ranking/graph+dbms (дата обращения: 22.08.2019).
18. Габриелян Г.А. Графовая база данных NEO4J для проектирования высоконагруженных систем. Студенческий электрон. научн. журн. 2018. № 11(31). URL: https://sibac.info/journal/student/31/111409. (дата обращения: 12.08.2019).
19. JanusGraph. URL: https://janusgraph.org/ (дата обращения: 18.08.2019).
20. Dgraph. URL: https://dgraph.io/ (дата обращения: 22.08.2019).
21. TigerGraph. URL: https://www.tigergraph.com/ (дата обращения: 27.08.2019).
22. AllegroGraph. URL: https://franz.com/agraph/allegrograph/ (дата обращения: 27.08.2019).

REFERENCES:

[1] National Research Council et al. Funding a revolution: Government support for computing research. National Academies Press, 1999.
[2] Understanding JSON Schema. URL: https://json-schema.org/understanding-json-schema/reference/type.html (accessed: 12.08.2019).
[3] Eifrem E. Why Graph Database Could Be Key to Addressing Financial Services Challenges URL: https://financialit.net/blog/financial-services/why-graph-database-could-be-key-addressing-financial-services-challenges (accessed: 15.08.2019).
[4] Plaksiy K., Nikiforov A., Miloslavskaya N. Applying Big Data Technologies to Detect Cases of Money Laundering and Counter Financing of Terrorism. Proceedings of 2018 6th International Conference on Future Internet of Things and Cloud (FiCloud2018). Barcelona (Spain), 6-8 August 2018. P. 70–77. DOI 10.1109/W-FiCloud.2018.00017
[5] Harari F. Graph Theory. – M.: Mir, 1973. – 296 s. (in Russian).
[6] Kasyanov, V.N., Information visualization based on graph models. Scientific visualization. V.N. Kasyanov, E.V. Kasyanova. M.: Nauka, 2014. V. 6. n. 1. P. 31–50 (in Russian).

[7] Drezewski R.; Sepielak J.; Filipkowski W. The Application of Social Network Analysis Algorithms in a System Supporting Money Laundering Detection. In Digital Investigation June 2012 9(1). P. 8–21. Elsevier Ltd.
[8] SAS Anti-Money Laundering (SAS AML). URL:http://www.tadviser.ru/index.php/%D0%9F%D1%80%D0%BE%D0%B4%D1%83%D0%BA%D1%82:SAS_Anti-Money_Laundering_ (SAS_AML) (accessed: 14.08.2019).
[9] Neo4j – Platform for connected data. URL: https://neo4j.com (accessed: 02.08.2019).
[10] DataStax Enterprise Graph Super-powering your data relationships. URL: https://www.datastax.com/products/datastax-enterprise-graph/ (accessed: 03.08.2019).
[11] Fernandes D., Bernardino J. Graph Databases Comparison: AllegroGraph, ArangoDB, InfiniteGraph, Neo4J, and OrientDB. DATA. 2018. P. 373–380.
[12] Robinson I., Webber J., Eifrem E. Graph databases. O'Reilly Media. 2013.
[13] Tabacchi M. E. et al. Designing Cognitive Cities. Designing Cognitive Cities. Springer, Cham, 2019. P. 3–27.
[14] Noel S. et al. Big-data architecture for cyber attack graphs representing security relationships in nosql graph databases. 2015.
[15] Qiu L. et al. Trusted computer system evaluation criteria. National Computer Security Center. 1985.
[16] AllegroGraph 6.6.0 Triple Attributes. URL: https://franz.com/agraph/support/documentation/current/triple-attributes.html. Access date: 02.08.2019.
[17] DB-Engines Ranking of Graph DBMS URL: https://db-engines.com/en/ranking/graph+dbms (accessed: 22.08.2019).
[18] Gabrielyan G.A. GRAPHIC DATABASE NEO4J FOR DESIGN OF HIGH-LOADED SYSTEMS. Student electronic scientific journal. 2018. № 11 (31). URL: https://sibac.info/journal/student/31/111409 (accessed: 08.12.2019) (in Russian).
[19] JanusGraph. URL: https://janusgraph.org/ (accessed: 18.08.2019).
[20] Dgraph. URL: https://dgraph.io/ (accessed: 22.08.2019).
[21] TigerGraph. URL: https://www.tigergraph.com/ (accessed: 27.08.2019).
[22] AllegroGraph. URL: https://franz.com/agraph/allegrograph/ (accessed: 27.08.2019).

Поступила в редакцию – 28 августа 2019 г. Окончательный вариант – 16 сентября 2019 г.
Received – September 28, 2019. The final version – September 16, 2019.