УДК 004.65
DOI: 10.15827/2311-6749.18.1.3

АНАЛИЗ СОВРЕМЕННЫХ ТЕХНОЛОГИЙ ХРАНЕНИЯ СВЕРХБОЛЬШИХ ОБЪЕМОВ ИНФОРМАЦИИ

Н.Е. Тимофеева, зав. лаб., timofeevane@yandex.ru; К.А. Дмитриева, программист, krismail95@gmail.com; И.Д. Сагаева, к.ф.-м.н., доцент, sagaevaid@gmail.com
(Саратовский национальный исследовательский государственный университет имени Н.Г. Чернышевского, ул. Астраханская, 83, г. Саратов, 410012, Россия)

В статье делается обзор существующих технологий и программных решений для хранения большого объема информации.

Выбор технологии хранения данных, главным образом, влияет на производительность БД и всей системы в целом. Несмотря на то, что подходы и модели хранения сверхбольших данных непрерывно совершенствуются, как для крупных компаний, так и для научных групп организация хранения остается крайне трудоемким процессом. В большинстве случаев различные технологии хранения данных предназначены для решения конкретных задач.

Цель настоящей работы – обобщить имеющийся опыт хранения сверхбольших данных, который мог бы стать основой для создания новых методов. Авторы рассматривают сильные и слабые стороны технологий, применяющихся для хранения и управления большими объемами данных: распределенные и параллельные БД, технология MapReduce, их особенности и области применения. Также делают сравнительный анализ возможностей существующих программных решений, реализующих эти модели. На основе анализа даны рекомендации, направленные на совместное использование реляционных и нереляционных моделей с целью объединения преимуществ обеих технологий.

Ключевые слова: *БД, распределенная БД, параллельная БД, СУБД, NoSQL, MapReduce.*

На современном этапе развития информационных технологий ведущую роль играют технологии, обеспечивающие обработку большого потока разнородных данных. Каждые два года мировой объем информации увеличивается почти в 2 раза, что приводит к необходимости разработки новых методов хранения и обработки данных.

Для управления информацией используют СУБД, предоставляющие пользователю удобный инструмент управления данными без знания особенностей их аппаратного хранения в вычислительной машине.

БД, хранящие астрономические, географические, научные, медицинские, экономические и другие данные, а также данные, полученные в ходе экспериментов, представляют собой класс сверхбольших данных. Количественное определение их объема постоянно меняется во времени и зависит от последних достижений технологий физического хранения информации. В связи с этим вопрос об улучшении, разработке и модификации методов хранения данных остается актуальным во все времена.

Выбор технологии хранения данных, главным образом, влияет на производительность БД и всей системы в целом. В работе [1] представлены результаты исследований производительности и надежности СУБД, но встал вопрос о хранении и обработке больших данных. В связи с этим возникла необходимость в проведении анализа различных подходов к хранению данных с учетом требований конкретных задач.

Данная статья посвящена анализу технологий и технологических решений хранения большого объема информации, принимая во внимание особенности их применения.

Основные технологии хранения большого объема данных

Одной из необходимых составляющих компьютера является память. Все современные вычислительные машины оснащены аппаратным обеспечением для обработки и хранения информации. В наше время невозможно представить работу приложения без получения, обработки и записи определенного типа данных. Для хранения информации используют БД, которые, в свою очередь, работают через ПО, называемое СУБД.

В связи с увеличением объемов информации локальной БД становится недостаточно и тогда используют распределенные и параллельные БД, а также довольно молодую технологию MapReduce. Рассмотрим основные понятия и отличия этих технологий.

Распределенная БД (РБД) – это совокупность множества взаимосвязанных БД, распределенных в компьютерной сети. РБД, в свою очередь, состоит из *узлов приема запросов*, на которых реализован пользовательский интерфейс для доступа к данным, и *узлов данных*, на которых, собственно, хранятся сами данные. *Система управления РБД* (РСУБД) организует прозрачный доступ для пользователя, то есть РБД для пользователя и прикладных программ выглядит как локальная БД [2, 3].

Стоит заметить, что узлы логически представляют собой независимые компьютеры, которые, в свою очередь, могут иметь различное переменное окружение [3], то есть узлы РБД – компьютеры, связанные коммуникативной сетью, а не процессоры, входящие в многопроцессорную систему.

Для распределения данных по узлам сети используют два механизма: ***репликация*** и ***фрагментация*** данных.

Репликацией называют механизм синхронизации копий фрагментов данных, который используют для увеличения производительности и доступности. Увеличение производительности достигается за счет экономии затрат на пересылку данных между узлами, однако при увеличении копий фрагментов увеличиваются и затраты на обновление данных. Таким образом, поиск оптимального размещения данных представляет сложную, индивидуальную для каждой системы задачу оптимизации [2–4].

Фрагментация данных, или *шардинг* – это разделение переменного отношения на фрагменты и их хранение на разных узлах. Фрагментируют данные на горизонтальные или вертикальные разделы. *Горизонтальная фрагментация* реализуется при помощи операции селекции, *вертикальная фрагментация* – при помощи проекции [5]. Также стоит заметить, что информация о местонахождении каждого фрагмента должна храниться в глобальном словаре, который, в свою очередь, может быть централизованным или распределенным. Фрагментация повышает производительность системы за счет хранения фрагментов там, где они чаще используются [2–4].

На архитектурные особенности ее построения влияют также степень однородности РБД и ее архитектура. Наиболее популярной в настоящее время является архитектура ***клиент-сервер*** [2, 3, 6].

Параллельная БД – это БД, которая находится под управлением СУБД, реализованной для многопроцессорного компьютера [3, 7, 8].

Заметим, что подходы при построении параллельной БД направлены на более полное использование преимуществ конкретного мультипроцессора, тем самым утрачивая свойство переносимости [3]. Таким образом, параллельные БД являются аппаратно-программными комплексами.

Для обработки данных существуют три основных вида параллелизма. ***Межзапросный*** параллелизм предполагает одновременное выполнение множества запросов, относящихся к разным транзакциям. Под ***внутризапросным*** параллелизмом понимается одновременное выполнение нескольких операций, относящихся к одному и тому же запросу. Понятие ***внутриоперационного*** параллелизма означает параллельное выполнение одной операции в виде набора субопераций с применением в дополнение к фрагментации данных и фрагментации функций [3].

Архитектуру параллельных машин БД принято разбивать на три класса: архитектуры с разделяемой памятью и дисками, архитектуры с разделяемыми дисками и архитектуры без совместного использования ресурсов [9].

В системах с ***разделяемой памятью и дисками (SE-система)*** все процессоры при помощи общей шины соединяются с разделяемой памятью и дисками. При небольших конфигурациях SE-системы показывают более высокую производительность, однако данные системы имеют ограниченную масштабируемость (не более 20–30 процессоров) и низкую аппаратную отказоустойчивость [3, 9, 10].

Системы с ***разделяемыми дисками (SD-система)*** определяются следующим образом: у каждого процессора имеются своя оперативная память и общее дисковое пространство. SD-архитектура по сравнению с SE-архитектурой демонстрирует лучшую масштабируемость и более высокую степень отказоустойчивости, однако при ее реализации возникает ряд технических проблем и в настоящее время она не используется в чистом виде [3, 9, 10].

В системах ***без совместного использования ресурсов (SN-система)*** каждый процессор имеет собственную память и собственный диск. Таким образом, каждый узел SN-системы можно грубо рассматривать как локальную машину в РБД, разница между SN-системой и распределенными СУБД сводится к различию платформ реализации. Архитектуры без разделяемых ресурсов обладают следующими преимуществами: низкие затраты, расширяемость, высокая доступность, однако основными проблемами являются сложность реализации и балансировка нагрузки [3, 9, 10].

Новым направлением в обработке больших данных является модель ***MapReduce***, разработанная компанией Google в 2004 г. Работа MapReduce состоит из двух основных шагов: *Map* и *Reduce*. На стадии *Map* происходят предварительная обработка и фильтрация исходных данных. Функция Map применяется ко всем входным данным и на выходе выдает множество пар ключ–значение [11–13]. Затем следует стадия *Reduce*. На ней происходит свертка обработанных данных, на основе которых формируется результат, некий аналог выборки в реляционной БД. Между стадиями Map и Reduce иногда выделяются несколько промежуточных шагов, на которых осуществляют дополнительные операции над данными. Например, автор работы [11] выделяет промежуточную стадию *Shuffle*, на которой происходит разбиение выходных данных стадии *Map* на секции для последующего применения к ним алгоритмов распараллеливания.

Преимуществом MapReduce является возможность распределенно производить операции предварительной обработки и свертки. Таким образом, данная модель применяется для обработки больших объе-

мов неструктурированных, сырых данных, однако она не подходит для обработки данных в реальном времени [14]. На данный момент активно разрабатываются модели внедрения технологии MapReduce в традиционные реляционные СУБД [15, 16] для сохранения преимуществ обеих парадигм.

Свободно распространяемые решения для хранения большого объема информации

На рынке в данный момент существуют коммерческие решения, которые позволяют обрабатывать большие объемы данных, но они в основном ориентированы на ту или иную специфическую аппаратную платформу (DB2 Parallel Edition, NonStop SQL, NCR Teradata, Oracle RAC, Greenplum, Microsoft SQL Server и др.). Кроме того, коммерческие решения являются дорогостоящими, что ограничивает их массовое использование.

Однако сейчас активно разрабатываются и внедряются методы и расширения для свободно распространяемых СУБД с открытым исходным кодом, которые обеспечивают параллельную обработку запросов и транзакций. Каждая из СУБД реализует одну из моделей данных: реляционную и нереляционную (NoSQL). Эти модели являются главным критерием того, как будет работать и управлять информацией приложение. На основе реляционных СУБД разрабатываются кластерные решения, а на основе NoSQL получают альтернативные. Также на данный момент активно развивается направление гибридного использования двух моделей, сочетающее необходимые свойства одной и другой моделей.

Реляционные решения. Данный класс представляет собой решения, основанные на реляционной модели данных. Далее названы наиболее популярные свободно распространяемые СУБД для этого условного класса.

MySQL Cluster [17, 18] представляет собой совокупность серверов хранилищ данных и серверов управления. MySQL Cluster предназначен для построения распределенной архитектуры БД без единой точки отказа. Свободная параллельная СУБД MySQL Cluster обладает хорошей производительностью и масштабируемостью на простых запросах, но эффективность СУБД падает при обработке сложных запросов [19–21].

Другой популярной реализацией кластера БД на основе свободно распространяемой реляционной СУБД MySQL является ***Galera Cluster*** [22], которая, в свою очередь, поставляется двумя производителями как свободно распространяемые продукты *Percona XtraDB Cluster* [23] и *MariaDB Galera Cluster* [24]. Galera позволяет создавать Multi-Master кластер БД. В реализации MariaDB акцент делается на оптимизацию запросов и расширение возможностей MySQL. В XtraDB – на увеличение производительности, доступности и масштабируемости для БД с большой пропускной способностью. В работах [25, 26] был осуществлен сравнительный обзор кластерных решений, за основу которых взята СУБД MySQL, показывающий преимущества и недостатки рассматриваемых реализаций.

Еще одной реляционной СУБД является свободно распространяемая СУБД **PostgreSQL** [27], в которой разработчики внедрили возможности распараллеливания запросов [28] и кластеризации БД с использованием родного или стороннего инструмента [29].

Также существует кластерная реализация **Postgres-XL** [30]. СУБД Postgres-XL – это массивно-параллельная горизонтально масштабируемая БД. Используется данный продукт прежде всего для бизнес-аналитики и хранения больших данных. Основными возможностями Postgres-XL являются массово-параллельная обработка запросов, распределенная обработка транзакций, масштабируемость и высокая доступность данных [31].

Основная идея ***ParGRES*** [32] во внедрении механизмов параллельной обработки запросов непосредственно в код свободно распространяемой СУБД PostgreSQL. СУБД ParGRES представляет собой промежуточное ПО, которое управляет экземплярами свободной последовательной СУБД PostgreSQL и запускается на узлах кластерной системы. Данную СУБД применяют при решении задач класса OLTP (обработка транзакций в реальном времени) [33, 34].

В рамках проекта «ОМЕГА. Разработка параллельной СУБД для мультипроцессорных вычислительных систем с кластерной архитектурой» [32] разрабатывается прототип параллельной СУБД «***Омега***». Проект направлен на оптимизацию работы управляющего узла, процесс-«координатор» которого отвечает за обработку запросов и балансировку нагрузки [35].

БД **NoSQL** предлагают новые функции, которые отсутствуют в традиционных реляционных СУБД; например, они позволяют хранить простые пары ключ–значение, сохранять неструктурированные коллекции данных и т.п. БД NoSQL в зависимости от целей и функционала подразделяют на четыре операционные модели: хранилище «ключ–значение», хранилище колонок или распределенное хранилище, документо-ориентированные СУБД, а также СУБД, основанные на графах [36–39].

Для хранения больших объемов данных используют хранилище колонок. В связи с этим рассмотрим NoSQL СУБД, предназначенные для распределенного хранения данных.

Apache Cassandra [40] – распределенная NoSQL СУБД, рассчитанная на создание высокомасштабируемых и надежных хранилищ данных, представленных в виде хэша. Cassandra получила большое рас-

пространение в производственной среде за счет экономически эффективного развертывания БД без увеличения количества узлов. Cassandra имеет линейную масштабируемость и высокую отказоустойчивость, а также более высокие показатели хранения данных по сравнению с реляционными СУБД. Выполнение запросов на небольших данных работает медленнее, чем реляционные СУБД, однако при достаточно больших данных Cassandra работает значительно быстрее [41].

Hbase [42] – NoSQL-распределенная СУБД с открытым исходным кодом, используемая в Apache Hadoop. Работает поверх распределенной файловой системы HDFS и обеспечивает отказоустойчивый способ хранения больших объемов разреженных данных. HBase позволяет в реальном времени осуществлять доступ к операции чтение/запись больших данных, имеет линейную и модульную масштабируемость. Данная СУБД обеспечивает лучшее решение для высокопроизводительной записи и хранения геопространственных данных по сравнению с реляционными СУБД [43]. HBase неэффективно работает с многомерными запросами, однако, если применить к ней модель данных CFIDM, производительность многомерных запросов увеличивает в 5х раз [44].

Специфические решения либо предназначены для очень узкого класса задач и их трудно отнести к какой-либо рассматриваемой модели, либо являются гибридами моделей SQL и NoSQL.

SciDB [45] – это свободно распространяемая постреляционная СУБД, ориентированная на обработку научных данных. Система оптимизирована для единовременной записи малоструктурированных данных и их последующего интенсивного чтения. Хранение данных в SciDB организовано в виде многомерных вложенных массивов, для обработки которых используют языки AQL (Array Query Language) и AFL (Array Functional Language) [46].

Hadoop [47] – это проект с открытым исходным кодом. Используется для надежных, масштабируемых и распределенных вычислений, а также применяется как хранилище файлов общего назначения. Hadoop состоит из двух ключевых компонентов: распределенная файловая система (HDFS) и система MapReduce, предназначенная для вычислений и обработки больших объемов данных на кластере. Hadoop вместе с СУБД Hive, Impala, Shark, Spark SQL, Drill для реляционного решения или HBase для NoSQL-решения позволяют работать с большими объемами данных, выполняя их распределенную обработку. Однако у Hadoop есть серьезные ограничения: невозможно внести изменения в данные, хранящиеся в системе HDFS, и Hadoop не поддерживает транзакции [15, 48].

Проект ***HodoopDB*** [49] – это гибридная система, включающая в себе преимущества параллельных БД, у которых хорошо развито свойство производительности, и платформы Map Reduce, у которой, в свою очередь, хорошо развиты свойства отказоустойчивости и работоспособности в неоднородной среде. Проект HodoopDB является свободно распространяемым продуктом, который использует на уровне управления БД СУБД PostgreSQL или СУБД MySQL, Hadoop – на уровне коммуникаций, СУБД Hive – на уровне компиляции. Основная идея HadoopDB состоит в связывании нескольких одноузловых систем БД с использованием Hadoop в качестве координатора задач и сетевого коммуникационного слоя. При отсутствии отказов производительность HadoopDB может приблизиться к производительности параллельных систем БД. Однако достичь той же или более высокой производительности параллельной СУБД HodoopDB не имеет возможности ввиду ограничений, которые связаны с СУБД, используемыми на узлах, а также с ограничениями платформы Hadoop [50].

Заключение

В данной статье был проведен анализ технологий и программных решений для хранения большого объема информации. Авторами рассмотрены используемые в современном мире подходы к хранению сверхбольших данных, а именно параллельные и распределенные БД, парадигма MapReduce.

Параллельные БД представляют собой программно-аппаратные комплексы и нуждаются в специализированном аппаратном обеспечении. Распределенные БД и NoSQL БД, построенные с использованием технологии MapReduce, строятся в вычислительных сетях, которые, в свою очередь, могут иметь разное операционное окружение.

Распределенные и параллельные БД хранят данные согласно реляционной модели. Работы по улучшении масштабируемости и отказоустойчивости распределенных и параллельных БД [51–53] ведутся, но они все еще уступают решениям, построенным на основе NoSQL БД. NoSQL БД, напротив, хранят данные в неструктурированном или слабоструктурированном виде. За счет такого хранения данных достигаются более высокая производительность и горизонтальная масштабируемость системы, однако при этом теряются преимущества распределенной модели, например, консистентность данных.

Таким образом, выбор модели хранения данных необходимо делать, отталкиваясь от конкретных задач и возможностей. Например, для обработки данных в реальном времени целесообразно использовать кластерные решения параллельных/распределенных БД. А NoSQL-решения предназначены для обработки большого потока неструктурированных данных и для систем, ориентированных на дальнейшее масштабирование. Стоит отметить, что на данном этапе развития технологий хранения данных ведутся ра-

боты по совместному использованию реляционной модели и парадигмы MapReduce, тем самым объединяя преимущества обеих технологий.

Литература

1. Тимофеева Н.Е., Гераськин А.С., Полулях К.А. Исследование и построение моделей нагрузочного тестирования СУБД для повышения скорости и производительности распределенной вычислительной системы // Вестн. Волгоградского гос. ун-та: Математика. Физика. 2017. № 1. С. 75–89.
2. Дейт К. Дж. Введение в системы баз данных; [пер. с англ.]. М.: Вильямс, 2005. 1328 с.
3. Özsu M.T. and Valduriez P. Distributed and parallel database systems. In Handbook of Computer Science and Engineering. Tucker A. (ed.), CRC Press, 1997, pp. 1093–1111.
4. Бойченко А.В., Рогожин Д.К., Корнеев Д.Г. Алгоритм динамического масштабирования реляционных баз данных в облачных средах // Статистика и экономика. 2014. № 6. Т. 2. С. 461–465.
5. Пушников А.Ю. Введение в системы управления базами данных. Ч. 1: Реляционная модель данных. Уфа: Изд-во Башкирского ун-та, 1999. 108 с.
6. Кузнецов С.Д. Основы баз данных. М.: Бином, 2007. 488 с.
7. DeWitt D. and Gray J. Parallel database systems: the future of high-performance database systems. Communications of ACM, 1992, no. 35, vol. 6, pp. 85–98.
8. Барон Г.Г. Параллельные архитектуры серверов баз данных. URL: https://www.osp.ru/data/www2/dbms/1995/02/22.htm (дата обращения: 12.11.2017).
9. Stonebraker M. The case for shared nothing. Database Engineering Bulletin, 1986, no. 1, pp. 4–9.
10. Соколинский Л.Б. Параллельные машины баз данных // Природа. 2001. № 8. С. 10–17.
11. Гладкий М.В. Модель распределенных вычислений MapReduce // Тр. БГТУ: Физ.-мат. науки и информ. 2016. № 6. С. 194–198.
12. Клеменков П.А., Кузнецов С.Д. Большие данные: современные подходы к хранению и обработке // Тр. ИСП РАН. 2012. Т. 23. С. 143–158.
13. Янишевская А.Г., Чурсин М.А. Способы хранения и обработки большого объема данных с использованием MapReduce и Percona Server // Инженерный вестн. Дона. 2015. № 2. Ч. 2. Т. 36. URL: ivdon.ru/ru/magazine/archive/n2p2y2015/ 2976 (дата обращения: 12.11.2017).
14. Pavlo A., Paulson E., Rasin A., etc. A comparison of approaches to large-scale data analysis. Proc. 35th SIGMOD Intern. Conf. on Management of Data. USA, 2009, pp. 165–178.
15. Кузнецов С.Д. MapReduce: внутри, снаружи или сбоку от параллельных СУБД? // Тр. ИСП РАН. 2010. Т. 19. С. 35–70.
16. Дергачев А.А. Анализ данных на основе плат формы SQL-mapreduce // Науч.-технич. вестн. информ. технологий, механики и оптики. 2014. № 1. С. 66–71.
17. MySQL Cluster CGE // MySQL official product website. URL: https://www.mysql.com/products/cluster/ (дата обращения: 18.11.2017).
18. Oracle. MySQL cluster evaluation guide. Technical White Paper, 2013. URL: http://omegaegitim.com/sources/mysql_wp_cluster_evalguide.pdf (дата обращения: 18.11.2017).
19. Ammar Fuad, Alva Erwin, Heru Purnomo Ipung. Processing performance on Apache Pig, Apache Hive and MySQL cluster. Proc. Intern. Conf. ICTS 2014, Surabaya, Indonesia, 2014, pp. 297–301.
20. Franke C., Morin S., Chebotko A., Abraham J., Brazier P. Efficient Processing of Semantic Web Queries in HBase and MySQL Cluster. IT Professional, 2013, vol. 15, no. 3, pp. 36–43.
21. Hubel M. Technical Comparison of DB2 and MySQL. Martin Hubel Consulting Inc., 2004, 32 p.
22. SyCraft «Идеальный» кластер. Ч. 3.1. Внедрение MySQL Multi-Master кластера. 2015. URL: https://habrahabr.ru/post/253869 (дата обращения: 18.11.2017).
23. Percona XtraDB Cluster. URL: https://www.percona.com/software/mysql-database/percona-xtradb-cluster (дата обращения: 18.11.2017).
24. MariaDB Galera Cluster. URL: https://downloads.mariadb.org/mariadb-galera/org/mariadb-galera/ (дата обращения: 18.11.2017).
25. MySQL Vs. MariaDB Vs. Percona. URL: https://www.atlantic.net/community/whatis/mysql-vs-mariadb-vs-percona/ (дата обращения: 18.11.2017).
26. System Properties Comparison MariaDB vs. MySQL vs. Percona Server Percona. URL: https://db-engines.com/en/system/MariaDB%3BMySQL%3BPercona+Server (дата обращения: 18.11.2017).
27. PostgreSQL. URL: https://www.postgresql.org/ (дата обращения: 20.11.2017).
28. Canovai F. PostgreSQL 9.6: Parallel Sequential Scan. 2016. URL: https://blog.2ndquadrant.com/postgresql96-parallel-sequential-scan/ (дата обращения: 20.11.2017).

29. PostgreSQL кластер и заметки по работе с 1С сервером. URL: https://itisok.ru/stati/postgresql-klaster-i-zametki-po-rabote-s-1s-serverom (дата обращения: 20.11.2017).

30. Postgres-Xl. URL: https://www.postgres-xl.org/ (дата обращения: 18.11.2017).

31. Postgres-XL. 2016 URL: https://www.2ndquadrant.com/en/resources/postgres-xl/ (дата обращения: 20.11.2017).

32. Проект ОМЕГА. Разработка параллельной СУБД для мультипроцессорных вычислительных систем с кластерной архитектурой. URL: http://omega.susu.ru/ (дата обращения: 20.11.2017).

33. Пан К.С., Соколинский Л.Б., Цымблер М.Л. Интеграция параллелизма в СУБД с открытым кодом // Открытые системы. СУБД. 2013. № 9. С. 56–58.

34. Пан К.С., Цымблер М.Л. Разработка параллельной СУБД на основе последовательной СУБД PostgreSQL с открытым исходным кодом // Вестн. ЮУрГУ: Математическое моделирование и программирование. 2012. № 18. С. 112–120.

35. Гавриш Е.В., Колтаков А.В., Медведев А.А., Соколинский Л.Б. Параллельная СУБД с открытым исходным кодом для кластерных вычислительных систем // Вестн. ЮУрГУ: Вычислительная математика и информатика. 2013. № 3. С. 80–91.

36. Amber Модели и системы управления базами данных NoSQL. 2016. URL: https://www.8host.com/blog/modeli-i-sistemy-upravleniya-bazami-dannyx-nosql/ (дата обращения: 22.11.2017).

37. Сравнение NoSQL систем управления базами данных. URL: http://devacademy.ru/posts/nosql/ (дата обращения: 22.11.2017).

38. Deka Ganesh Chandra. A survey of cloud database systems.2014. URL: http://romisatriawahono.net/lecture/rm/survey/network%20security/Deka%20-%20Survey%20of%20Cloud%20Database%20Systems%20-%202013.pdf (дата обращения: 22.11.2017).

39. Мухина Ю.Р. Обзор noSQL решений управления данными // Управление в современных системах. 2013. № 1. С. 68–73.

40. Apache CASSANDRA. URL: http://cassandra.apache.org/ (дата обращения: 22.11.2017).

41. Muh. Rafif Murazza, Arif Nurwidyantoro. Cassandra and SQL database comparison for near real-time Twitter data warehouse. Proc. Conf. ISITIA, Lombok, Indonesia. 2016, no. 7, vol. 28, pp. 195–200.

42. Apache HBase. URL: http://hbase.apache.org/ (дата обращения: 22.11.2017).

43. Fan Gao, Peng Yue, Zhaoyan Wu, Mingda Zhang. Geospatial data storage based on HBase and MapReduce. Agro-Geoinformatics. Proc. 6th Intern. Conf., Fairfax, Virginia, USA, 2017, pp. 55–59.

44. Chun Cao, Weiyi Wang, Ying Zhang, Xiaoxing Ma. Leveraging Column Family to Improve Multidimensional Query Performance in HBase. Cloud Computing (CLOUD). Proc. 10th Intern. Conf., Honolulu, CA, USA, 2017, pp. 106–113.

45. SciDB. URL: http://www.paradigm4.com/try_scidb/ (дата обращения: 22.11.2017).

46. Бартунов О., Велихов П. SciDB: СУБД для научных данных // Научный сервис в сети Интернет: экзафлопсное будущее: тр. Междунар. науч. конф. М.: Изд-во МГУ, 2011. С. 427–431.

47. Apache. Hadoop. URL: http://hadoop.apache.org/ (дата обращения: 22.11.2017).

48. Гусейнов А.А., Бочкова И.А. Исследование распределенной обработки данных на примере системы Hadoop // Актуальные проблемы авиации и космонавтики. 2016. № 12. С. 606–608.

49. HadoopDB. URL: http://db.cs.yale.edu/hadoopdb/hadoopdb.html (дата обращения: 22.11.2017).

50. Abouzeid A., Bajda-Pawlikowski K., Abadi D., Silberschatz A., Rasin A. HadoopDB: an architectural hybrid of mapreduce and DBMS technologies for analytical workloads. Proc. 35th VLDB Conf., France, 2009, no. 2, pp. 922–933.

51. Sukheja Deepak, Singh Umesh Kumar, Mishra Durgesh, Pandya Bhupendra K. Query processing and optimization of parallel database system in multi-processor environments. Proc. Asia Intern. Conf. Modelling Symposium (AMS). 2012, pp. 191–194.

52. Gibadullin R.F., Vershinin I.S., Minyazev R.Sh. Realization of replication mechanism in PostgreSQL DBMS. Proc. 2017 Intern. Conf. Industrial Engineering, Applications and Manufacturing (ICIEAM-2017), St. Petersburg, 2017, pp. 1–6.

53. Asma H. Al-Sanhani, Amira Hamdan, Ali B. Al-Thaher, Ali Al-Dahoud. A comparative analysis of data fragmentation in distributed database. Proc 8th Intern. Conf. Inform. Tech. (ICIT), 2017, pp. 724–729.