электронный журнал
МОЛОДЕЖНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51038.

УДК 004.75

Развитие интерфейсов обмена данными между современными высоко параллельными приложениями и СУБД

***Тихонов И.В.**, студент*
кафедра «Системы обработки информации и управления»
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана

Научный руководитель: Сухобоков А.А., к.т.н., доцент
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана
bauman@bmstu.ru

1. Современные высоко параллельные системы управления базами данных (СУБД). Современные СУБД в значительной мере вышли за границы функциональности, ранее обозначаемой этим термином. Для них часто стал использоваться термин «платформы приложений» или просто «платформы», когда понятно, что речь идёт о программном обеспечении, а не об аппаратуре. Платформы основываются на ряде технологий, обеспечивающих параллельную обработку структурированных и неструктурированных данных. В число этих технологий входят:

- стандартные технологии реляционных СУБД и средства работы с SQL;
- технологии, обеспечивающие тесную интеграцию с интернет-приложениями;
- технологии работы с таблицами по столбцам;
- технологии работы с текстами и поиска в текстах;
- технологии проведения всех операций непосредственно в оперативной памяти (так называемые In-Memory СУБД);
- технологии сжатия данных;
- технологии резервного копирования всех изменений на диски с целью обеспечения возможности восстановления при сбоях;
- средства и языки разработки приложений;
- различные инструменты получения данных из различных источников, загрузки их в базу данных и выгрузки из неё.

Одним из наиболее распространённых примеров современной платформы, широко используемой для создания приложений, является SAP HANA (High-performance Analytic

Appliance). SAP HANA - это высокопроизводительный инструмент для аналитики разработанный компанией SAP и официально анонсированный в 2010 году. На рынок SAP HANA была выведена в 2011 году, после объявления официальных результатов тестирования. Общая архитектура SAP HANA была представлена в [1] в виде показанном на рис.1

Рис. 1. Общая архитектура SAP HANA

Высокая степень параллелизма обработки данных в SAP HANA обеспечивается техническими решениями по распределению таблиц данных, отдельных столбцов и фрагментов столбцов между серверами, процессорами и процессорными ядрами, образующими аппаратную среду для работы SAP HANA. Эти решения [2] показаны на рис. 2 и рис. 3.

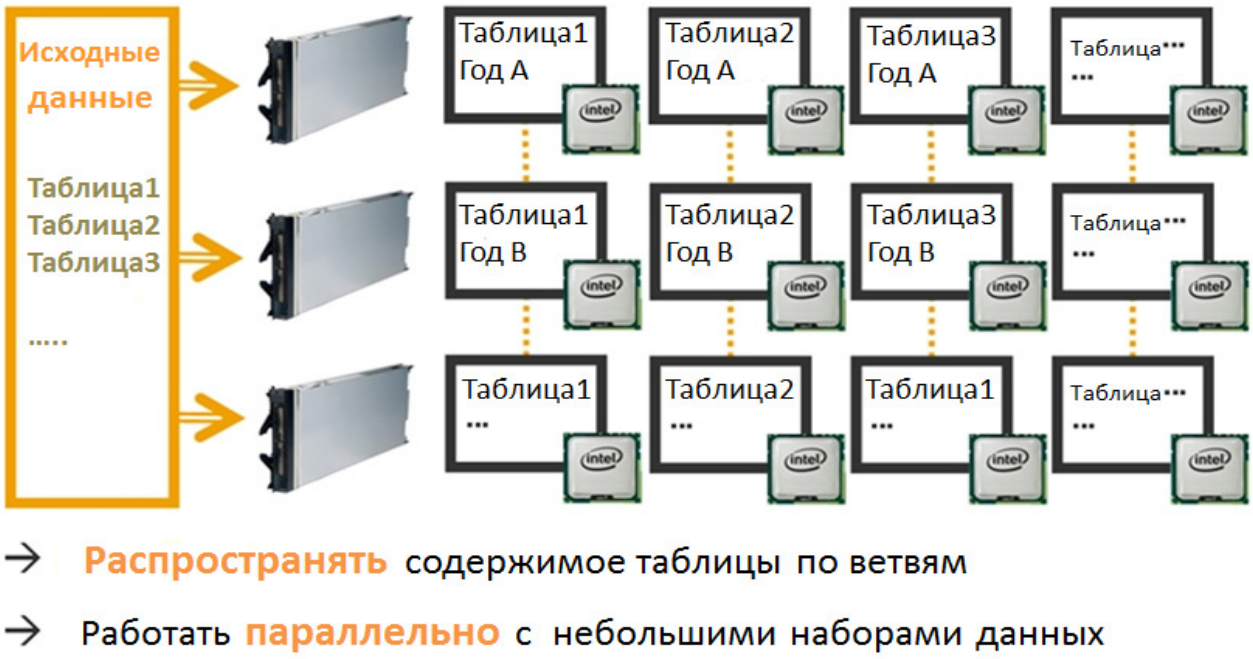

Рис. 2. Распределение обработки таблиц между серверами и процессорами по группам строк

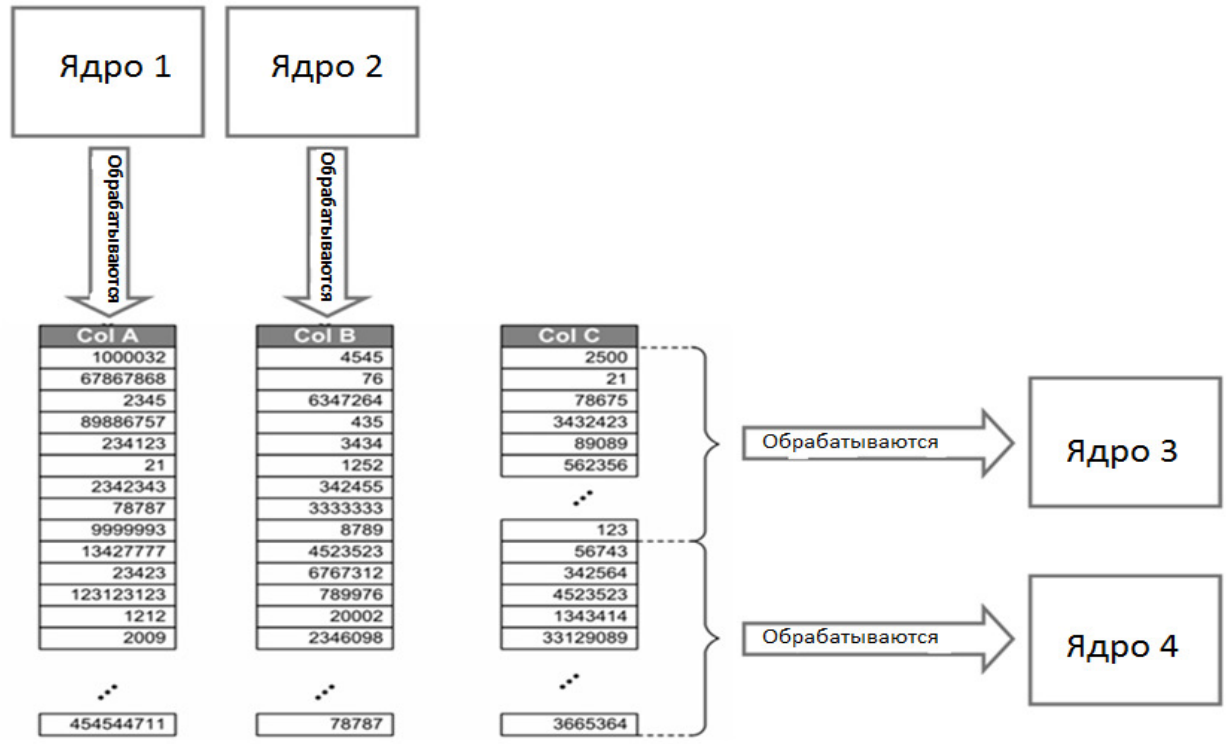

Рис. 3. Распределение обработки столбцов одной таблицы между процессорными ядрами

2. Кластеры Hadoop как основа для создания высоко параллельных приложений. Начало разработки Hadoop относится к 2002 году [3]. Автор Hadoop - Дуг Каттинг. Идеи Hadoop и первая реализация появились у него во время работы над проектом Nutch (система веб-поиска с открытым кодом). В 2004 году команда разработчиков Hadoop начала реализацию системы с открытым кодом — NDFS (Nutch Distributed File System), по подобию распределенной файловой системы компании Google GFS (Google File System), поскольку она могла справиться с задачей хранения больших файлов, генерируемых при обходе и индексировании сайтов. В 2004 Google представила технологию MapReduce. В 2005 году разработчики Nutch создали полноценную реализацию MapReduce на базе Nutch, чуть позже они адаптировали все алгоритмы для использования MapReduce и NDFS. В 2008 году Hadoop стал одним из ведущих проектов Apache, к тому времени он уже был внедрен и успешно использовался в таких компаниях, как Yahoo!, Facebook , Last.Fm и. т. д. Сегодня Hadoop используется многими компаниями.

Hadoop – это набор утилит, библиотек, и программный каркас, предназначенный для разработки и выполнения распределённых программ, работающих на кластерах из сотен и тысяч узлов. Используется для реализации контекстных и поисковых механизмов многих высоконагруженных веб-сайтов, а также в приложениях Big Data.

Для приложений, построенных с использованием Hadoop, характерны следующие черты [4]:

- использование кластера стандартных серверов;
- использование модели программирования Hadoop MapReduce;
- применение архитектуры программного обеспечения Hadoop;
- применение решений и инструментов, входящих в экосистему Hadoop.

Кластер стандартных серверов Hadoop состоит из стандартных (недорогих) серверов. Самые большие существующие кластеры включают более 40 000 узлов (серверов). Большинство серверов, в терминологии Hadoop, "DataNode" (рабочие узлы), содержат только часть данных. Задания по обработке данных разбиваются на сотни и тысячи меньших заданий, и они выполняются параллельно на отдельных устройствах. Hadoop может обрабатывать петабайты или более хранящихся данных просто за счёт большого количества DataNode. Hadoop также обеспечивает автоматическую репликацию данных между DataNode, что означает, что если один из DataNode перестанет работать, то работа Hadoop не прервется, а потерянные данные будут восстановлены.

Текущие версии Hadoop имеют один "NameNode" (Главный узел), который управляет данными, хранящимися на DataNode, и данными репликации. NameNode также инициирует и управляет аналитикой и процессом работы и перезапускает DataNode, если происходит любой сбой, гарантируя, что работа Hadoop будет выполнена.

Модель программирования MapReduce – это инструмент, который позволяет Hadoop выполнять аналитические и обрабатывающие задачи в режиме параллельной обработки.

MapReduce – это основа (Фреймворк) для вычисления некоторого набора задач с использованием большого числа компьютеров, образующих кластер и связанных между собой высокоскоростными каналами для передачи данных. Алгоритм MapReduce состоит из двух частей: Map и Reduce. Шаг Map выполняет предварительную обработку исходных данных. Для этого один из компьютеров (Главный узел) получает исходные данные, делит их на части и передает остальным компьютерам для предварительной обработки. Шаг Reduce выполняет свёртку предварительно обработанных данных. Главный узел получает ответы от остальных (рабочих узлов) и на основе предоставленных ими данных формируется результат – решение задачи, которая была изначально поставлена. Основное преимущество использования MapReduce заключается в применении стандартного решения для распараллеливания вычислений между узлами кластера.

Наиболее известными производителями и разработчиками Apache Hadoop считаются две достаточно молодые компании Cloudera и Hortonworks. Ниже приведено их краткое описание:

- Cloudera — американская компания, основанная в 2008 году[5], основная цель ее бизнеса - это коммерциализация проекта Hadoop. Компания является разработчиком связующего программного обеспечения. Эта компания выпустила коммерческую версию Apache Hadoop. Она также оказывает услуги по облачной обработке данных при помощи технологии Hadoop.

  Одноименный продукт компании Cloudera Hadoop - это полностью открытый дистрибутив, который создавался при участии разработчика Hadoop Каттинга. Cloudera Hadoop распространяется в двух вариантах, платном и бесплатном. Платный вариант называется Cloudera Enterprise.

  Cloudera тесно сотрудничает с Intel. Компания Intel владеет 18% акций Cloudera и постоянно инвестирует в нее деньги, путем увеличения своего пакета акций в этой компании. В связи с этим по прогнозам аналитиков в ближайшем

будущем Cloudera станет ведущим игроком среди фирм, развивающих экосистему Apache Hadoop.

- Hortonworks – американская компания[6], которая была основана в 2011 году двадцатью четырьмя инженерами Yahoo!. Компания утверждает, что у её сотрудников больше всего опыта в работе с Hadoop. Hortonworks Hadoop представляет собой полностью открытый дистрибутив. Компания Hortonworks в 2013 году объявила о доступности продукта Hortonworks Data Platform 1.3 (HDP) для Windows.

Представители Hortonworks сообщили, что HDP 1.3 для Windows с открытым исходным кодом представляет собой единственный дистрибутив на базе Apache Hadoop, сертифицированный для работы под управлением Windows Server 2008 R2 и Windows Server 2012. Он позволяет создавать и развертывать аналитические приложения в операционных системах Windows на основе Hadoop.

3. Причины совместного использования разных подходов к организации параллельных вычислений. Выбор технологий обработки данных для решения бизнес-проблем должен осуществляться с учётом множества факторов [7]:

- стоимости оборудования и программного обеспечения;
- наличия средств разработки;
- величины операционных расходов, связанных с осуществлением сервисного обслуживания (как собственными силами, так и с привлечением подрядчиков);
- обеспечения безопасности;
- обеспечения необходимого уровня доступности;
- обеспечения надёжного резервного копирования и восстановления.

Во многих случаях наилучшие характеристики достигаются при комбинировании двух или более технологий работы с данными.

В этой статье были рассмотрены две высоко параллельные платформы для организации обработки данных: SAP HANA и Hadoop. Между этими платформами имеется несколько основных различий. Для обработки данных Hadoop использует дешёвые, общедоступные сервера (commodity servers) чтобы справиться с объемом данных, измеряемым в петабайтах и, потенциально, в экзабайтах. Этот объём данных намного выше, чем диапазон 100 ТБ или меньше, в рамках которого работают SAP HANA и обычные реляционные СУБД. С другой стороны, Hadoop значительно медленнее, чем

обычная СУБД, и гораздо медленнее, чем SAP HANA, которой требуется несколько минут (редко – часов), для получения аналитических результатов. Тем не менее, с помощью Hadoop легче обрабатывать произвольные структуры данных и, как правило, значительно дешевле стоимость оборудования для хранения данных в расчёте на один терабайт.

Таблица 1

Сравнение характеристик приложений на платформах SAP HANA и Hadoop

In-Memory (SAP HANA®) Database	**Hadoop**
Главным образом структурированные данные в оперативной памяти	Любые данные любой файловой структуры на диске
Очень быстрый доступ к данным (<1мс)	Очень медленный доступ к данным (от одной минуты до нескольких часов)
Предварительно определенная структура хранения данных	Структура хранения данных не определена, или определяется в процессе обработки
Один или множество серверов (сотни ядер)	Распределенные серверы
Scale-up/scale-out архитектура	scale-out архитектура
Отказоустойчивость на уровне сервера	Отказоустойчивость на уровне сервера и на уровне запроса
Устройство базы данных $	Общедоступные (дешевые) сервера
Отличная обработка транзакций в режиме реального времени OLTP	Нет обработки транзакций в режиме реального времени OLTP
Отличная аналитическая обработка в режиме реального времени OLAP	Медленная аналитическая обработка в режиме реального времени OLAP
Высокая согласованность данных - на основе принципов ACID	Возможная несогласованность данных (BASE)
Все средства администрирования готовы для предприятия	Только несколько готовых средств администрирования для предприятий
Быстрый процесс инновации	Быстрый процесс инновации
Нехватка навыков в области ИТ	Нехватка навыков в области ИТ
Необходимо оплачивать лицензию	Никакой платы, открытые данные

Это означает, что Hadoop, в отличие от SAP HANA, не позволяет решать задачи в реальном времени (мгновенно). Однако он позволяет хранить и обрабатывать более объемные и детализированные данные с меньшей стоимостью, он позволяет выполнять более глубокий анализ особенностей бизнеса и работать с различными данными, описывающими задачу.

Сопоставление технологий SAP HANA и Hadoop [7] приводится в табл.1.

Технологии In-memory, реализованные в таких продуктах как SAP HANA лучше всего применять, когда важна скорость обработки данных. Например, для обновления и анализа данных в режиме реального времени, когда объем данных не чрезмерно большой и окупаются затраты на эту обработку.

Hadoop лучше подходит, если объем данных очень большой, типы данных сложны для обработки с помощью других технологий баз данных (например, имеется неструктурированный текст), и приемлем медленный анализ и обработка данных, или должны быть минимизированы расходы на эту обработку.

Поскольку каждая из платформ имеет свои преимущества и недостатки, можно совместно использовать SAP HANA и Hadoop. При совместном использовании SAP HANA и Hadoop появляется потенциал для очень быстрой обработки действительно больших данных.

4. Предыстория развития средств обмена данными между приложениями и СУБД. На протяжении всей истории развития корпоративных систем происходило расширение состава решаемых задач и функциональности систем. Одновременно шёл процесс выделения классов специализированных приложений наряду с возникновением коммерческих компаний, развивающих эти классы приложений и продвигающих их приложения на рынке. Одним из первых классов систем, которые получили широкое распространение, стали СУБД. При создании корпоративных систем предприятия стремятся использовать в своём ландшафте всё больше готовых приложений на базе СУБД и минимизировать объём уникальных разработок. Для построения корпоративных систем на базе готовых приложений, эти приложения должны иметь стандартизованные средства обмена данными с СУБД. Наличие такой стандартизации сокращает затраты на разработку и сопровождение приложений, а также позволяет минимизировать затраты при переходе с одной СУБД на другую.

4.1. ODBC. Одним из первых широко распространённых стандартов стал программный интерфейс доступа к базам данных ODBC API (Open Database Connectivity) [8], разработанный совместными усилиями Microsoft и Simba Technologies на основе спецификаций CLI (Call Level Interface), которые были предложены SQL Access Group, X/Open и Microsoft. Стандарт CLI был призван унифицировать программное взаимодействие с СУБД, сделать его независимым от поставщика СУБД и программно-аппаратной платформы. Впоследствии CLI был стандартизован ISO.

В начале 1990-х годов каждый из поставщиков баз данных предлагал собственный интерфейс. Если приложению было необходимо общаться с несколькими источниками данных, для взаимодействия с каждой из баз данных было необходимо написать свой код. С помощью ODBC прикладные программисты могли разрабатывать приложения с использованием единого интерфейса доступа к данным, не беспокоясь о тонкостях взаимодействия с разными СУБД.

4.2. JDBC. Промышленный стандарт JDBC (Java DataBase Connectivity) [9] для взаимодействия Java-приложений с различными СУБД, не зависящий от установленной операционной системы, был выпущен компанией Sun Microsystems 19-го февраля 1997 года как часть JDK в виде программного пакета java.sql, входящего в Java SE.

JDBC основан на концепции драйверов, позволяющих получать соединение с базой данных по специально описанному URL (Единый указатель ресурсов). JDBC API содержит как интерфейс для разработчиков приложений, так и интерфейс для разработчиков драйверов. Основное достоинство JDBC API состоит в том, что разработчику приложений, как и в случае с ODBC API не надо писать код под каждую СУБД.

5. Smart Data Access – средство обмена данными между высоко параллельными приложениями и высоко параллельными СУБД. Smart Data Access – это одно из стандартных средств обмена данными платформы SAP HANA, обеспечивающее интеграцию приложений с Hadoop. Smart Data Access осуществляет визуализацию данных для локального использования и использования в гибридной облачной среде. Smart Data Access отличается от других средств обмена данными SAP HANA, представляющих собой отдельные серверные системы (как, например, SAP Data Services, которое также обеспечивает обмен данными с Hadoop). Smart Data Access реализован в виде API, который можно рассматривать как дальнейшее развитие линии ODBC-JDBC.

Основные функциональные преимущества Smart Data Access:

- позволяет получать доступ к удаленным данным, с такой же скоростью как если бы они были локальными;
- обеспечивает интеллектуальную обработку запросов, включающую в себя разложение запросов на составные части с понижением предиката и необходимой функциональной компенсацией;

- не требует специального синтаксиса для доступа к различным источникам данных;
- поддерживает различные методы хранения данных.

Источники данных для Smart Data Access:

- Oracle, MS SQL, Teradata
- Hadoop / Hive (Hortonworks, Cloudera, Intel etc.)
- SAP Sybase ASE and IQ
- SAP Sybase ESP, SQLA

Обобщённая схема интерфейсов Smart Data Access представлена на рис.4

Рис. 4. Обобщённая схема интерфейсов Smart Data Access

Функциональные преимущества Smart Data Access обеспечиваются за счёт следующих факторов:

- Производительность и оптимизация запросов;
- Использование удаленных вычислительных систем;
- Мониторинг запросов и статистика;
- Мощные преобразования и проверка информации;

- «Умное» размещение и рекомендации по кэшированию;

Для поддержки функциональности Smart Data Access в его составе реализованы следующие компоненты [10]:

- Реляционные представления
- Трансформация типов данных
- Очистка и трансформация данных «на лету»
- Безопасность
- Оптимизация запросов
- Статистика данных (мониторинг данных)

Рис. 5. Высокоуровневая диаграмма компонентов Smart Data Access

- Кэширование данных
- Мониторинг запросов
- Механизм предсказаний (Recommendation Engine)

- Встраиваемые адаптеры
- Адаптер SDK

Общая схема взаимодействия компонентов Smart Data Access представлена на рис. 5 [10].

Общий перечень ключевых возможностей Smart Data Access представлен в табл.2.

Таблица 2

Ключевые возможности Smart Data Access

Характеристика	**Описание**
Источники данных	Oracle, MS SQL, IQ, ASE, Teradata, Hadoop/HIVE, HANA (BWoH, SoH)
Поддержка DDL	Select, Insert, Update, Delete
	Create/Drop Source (позволяет создавать источники данных)
	Create/Drop Virtual Tables (позволяет виртуальным таблицам, указывающим на удаленные таблицы, быть изменёнными и удалёнными)
Оптимизация	Проталкивание вниз фильтрации, проталкивание вниз агрегатов, сокращение полусоединений (Semi-Join reduction), перемещение соединений (join relocation), и т. д.
Исполнение	Функциональная компенсация
	Параллельное выполнение задач
	Функциональная трансляция
Безопасность	Доступ к удаленным источникам через вторичные учетные данные / или технического пользователя
Взаимодействие с адаптерами	Стандартный доступ к удаленным данным, доступ через ODBC адаптеры
	Преобразование типов данных удаленного сервера в типы данных HANA
Поддержка разработки	Создание / сброс источников данных / виртуальных таблиц
	Исполнение и мониторинг запросов
	Анализ планов запросов
	Поддержка генерируемых представлений для виртуальных таблиц

6. Выводы:

- Smart Data Access позволяет разработчикам приложений создавать новые приложения, интенсивно использующие данные, независимо от местоположения данных, их размера и содержания, а также функциональности создаваемых приложений.

- Наличие Smart Data Access усиливает возможности SAP HANA по обработке данных за счёт возможностей других платформ, таких как Hadoop, чтобы с высокой скоростью синтезировать необходимую для анализа информацию.
- Тем не менее, применение Smart Data Access не исключает необходимости создания и использования хранилищ данных, в которых обеспечивается целостность данных, процессы согласования и управления, требуемые для обеспечения соответствия данных другим источникам.
- Преимущества Smart Data Access для клиентов:
- быстрое развертывание высокопроизводительных приложений для интенсивной транзакционной и аналитической обработки данных;
- сокращение затрат;
- возможность использования преимуществ SAP HANA без переноса всех данных в HANA.

Список литературы

1. SAP HANA® Database for Next-Generation Business Applications and Real-Time Analytics. Explore and Analyze Vast Quantities of Data from Virtually Any Source at the Speed of Thought. Available at: http://download.sap.com/download.epd?context=E67AFF9FD4CFC6693FC443EB965A7B4FE599BA88ED6C9F622486D0E5BEB350EB7DAD751393D16A2D916A3C9EA834D1CB2A8138467760590E, accessed 01.04.2014).
2. HA100 SAP HANA Introduction // SAP AG, Course Version: 99, Participant Handbook, Material Number: 50119238. Valdorf. 2014. 329 p.
3. Тугов А. Hadoop. Часть 1: развертывание кластера. Режим доступа: http://blog.selectel.ru/hadoop-chast-1-razvertyvanie-klastera/ (дата обращения 1.04.2014)
4. Уайт Т. Hadoop. Подробное руководство. Санкт-Петербург: Издательство Питер, 2013. 672с.
5. Hadoop and Big Data. Available at: http://cloudera.com/content/cloudera/en/home.html, accessed 2.04.2014.
6. Why Hortonworks. Available at: http://hortonworks.com/hadoop/, accessed 3.04.2014)
7. CIO Guide. How to Use Hadoop with Your SAP® Software Landscape. Available at: http://www.sapbigdata.com/wp-content/uploads/2013/08/PATS_CIO-Guide_Hadoop-in-SAP-Landscape_final_May-22.pdf, accessed 03.04.2014.

8. Roger Sippl, SQL Access Group's Call-Level Interface. Available at: http://www.drdobbs.com/sql-access-groups-call-level-interface/184410032, accessed 01.04.2014.
9. Trail: JDBC(TM) Database Access. Available at: http://www.oracle.com/us/corporate/index.html, accessed 01.04.2014.
10. Balaji Krishna, SAP HANA Smart Data Access // Webinar of SAP Startup Focus Development Accelerator. Вальдорф. 2014. 24p.