УДК 004.65, 004.421 **DOI: 10.14529/cmse190203**

ОБЗОР МЕТОДОВ ИНТЕГРАЦИИ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА ДАННЫХ В СУБД

© 2019 М.Л. Цымблер

Южно-Уральский государственный университет
(454080 Челябинск, пр. им. В.И. Ленина, д. 76)
E-mail: mzym@susu.ru

Поступила в редакцию: 27.02.2019

Интеллектуальный анализ данных направлен на извлечение доступных для понимания знаний, необходимых для принятия решений в различных сферах человеческой деятельности. Феномен Больших данных является характерным признаком современного информационного общества. Процессы очистки и структурирования Больших данных приводят к образованию сверхбольших баз и хранилищ данных. Несмотря на появление большого количества NoSQL СУБД, основным инструментом управления базами данных по-прежнему остаются реляционные СУБД. Одним из перспективных направлений развития реляционных СУБД является внедрение в них средств интеллектуального анализа данных. Интеграция позволяет как избежать накладных расходов по экспорту анализируемых данных из хранилища и импорту результатов анализа обратно в хранилище, так и использовать при анализе данных системные сервисы, заложенные в архитектуре СУБД. В статье представлен обзор методов и подходов к решению задачи интеграции интеллектуального анализа данных в СУБД. Приводится классификация подходов к решению задачи интеграции интеллектуального анализа данных в СУБД. Представлены расширения языка баз данных SQL, обеспечивающие синтаксическую поддержку интеллектуального анализа данных в СУБД. Рассмотрены примеры реализации алгоритмов интеллектуального анализа данных на SQL и систем анализа данных в реляционных СУБД.

Ключевые слова: интеллектуальный анализ данных, реляционная СУБД, классификация, кластеризация, поиск шаблонов.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Цымблер М.Л. Обзор методов интеграции интеллектуального анализа данных в СУБД // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2019. Т. 8, № 2. С. 32–62. DOI: 10.14529/cmse190203.

Введение

В настоящее время феномен *Больших данных (Big Data)* оказывает существенное влияние на технологии обработки данных [3, 19]. На сегодня имеется широкий спектр приложений (социальные сети, электронные библиотеки, геоинформационные системы и др.), в которых производятся неструктурированные данные, имеющие сверхбольшие объемы и высокую скорость прироста (от 1 Тб в день). Исследования аналитической компании IDC показывают, что мировой объем данных удваивается каждые два года и к 2020 г. достигнет 44 Зеттабайт (44 трлн. Гб) [80].

В современном информационном обществе, однако, критичными являются не объемы и скорость прироста данных, а наличие эффективных методов и алгоритмов интеллектуального анализа данных. Под *интеллектуальным анализом данных (Data Mining)* понимают совокупность алгоритмов, методов и программного обеспечения для обнаружения в данных ранее неизвестных, нетривиальных, практически полезных и доступных интерпретации знаний, необходимых для принятия стратегически важных решений в различных сферах человеческой деятельности [25].

Процессы очистки и структурирования Больших данных приводят к образованию сверхбольших баз и хранилищ данных. Один из наиболее авторитетных ученых в области баз данных М. Стоунбрейкер указывает [75], что для решения проблем обработки сверхбольших данных необходимо использовать технологии систем управления базами данных (СУБД). Несмотря на появление большого количества NoSQL СУБД [22], СУБД на основе реляционной модели данных [21] по-прежнему остаются основным инструментом управления базами данных.

В 2016 г. в Бекманском отчете [6] ведущие мировые специалисты в области технологий обработки данных констатировали, что переход к умному обществу, управляемому данными, требует интегрированного и сквозного процесса от получения данных до извлечения из них полезных знаний.

Одним из перспективных направлений развития реляционных СУБД является внедрение в них средств интеллектуального анализа данных [54]. Размещение аналитических алгоритмов «рядом» с анализируемыми данными, имеющими сверхбольшие объемы, позволяет избежать существенных накладных расходов по экспорту анализируемых данных из хранилища и импорту результатов анализа обратно в хранилище [53]. Кроме того, интеллектуальный анализ данных внутри СУБД позволяет без дополнительных накладных расходов использовать системные сервисы, заложенные в архитектуре СУБД: отказоустойчивость, целостность и безопасность данных, исполнение запросов к данным на основе индексирования данных и управления буферным пулом и др.

Настоящая статья представляет собой обзор методов и подходов к решению задачи интеграции интеллектуального анализа данных в реляционные СУБД и организована следующим образом. В разделе 1 даны определения двух основных задач интеллектуального анализа данных: поиск шаблонов и кластеризация, — и кратко представлены основные алгоритмы их решения. В разделе 2 приводится классификация подходов к решению задачи интеграции интеллектуального анализа данных в СУБД. В разделе 3 рассмотрены методы реализации систем анализа данных в СУБД и представлены расширения языка баз данных SQL, обеспечивающие синтаксическую поддержку интеллектуального анализа данных в СУБД. Раздел 4 содержит примеры реализации алгоритмов интеллектуального анализа данных на SQL. Заключение резюмирует результаты, полученные в исследовании.

1. Основные задачи анализа данных и алгоритмы их решения

Поиск шаблонов и кластеризация являются одними из основных задач интеллектуального анализа данных [30]. Ниже приведены формальные определения указанных задач и кратко представлены алгоритмы их решения.

1.1. Поиск шаблонов

Поиск *шаблонов (pattern mining)* предполагает нахождение часто повторяющихся зависимостей в заданном наборе объектов и применяется в медицине (например, нахождение побочных эффектов лекарств), генной инженерии (поиск часто повторяющихся цепочек ДНК) и других предметных областях.

Общепринятой формой записи шаблона является *ассоциативное правило (association rule)*, которое формально определяется следующим образом. Пусть дано множество *объектов* $\mathcal{I} = \{i_1, i_2, \ldots, i_m\}$, любое непустое его подмножество называют *набором*.

Набор из k объектов ($1 \leqslant k \leqslant m$) называют *$k$-набором*. Пусть имеется множество *транзакций* (записей об операциях с объектами в данной предметной области) $\mathcal{D}$, в котором каждая транзакция представляет собой пару $(tid; I)$, где tid — уникальный идентификатор транзакции, $I \subseteq \mathcal{I}$ — набор. Ассоциативное правило представляет собой импликацию вида $A \to B$, где $A, B \subseteq \mathcal{I}$, $A \neq \emptyset, B \neq \emptyset$, $A \cap B = \emptyset$. Левая часть правила называется *антецедентом*, правая — *консеквентом*. В качестве мер полезности ассоциативных правил для аналитика применяются *поддержка (support)* и *достоверность (confidence)* правила, определяемые следующим образом:

$$\begin{aligned} support(A \to B) &:= P(A \cup B) \\ confidence(A \to B) &:= P(B \mid A). \end{aligned} \tag{1}$$

Устойчивым ассоциативным правилом (strong rule) называют правило, поддержка и достоверность которого не ниже наперед заданных пороговых значений *minsup* и *minconf* соответственно. Поиск устойчивых ассоциативных правил может быть разбит на две последовательно выполняемые задачи: поиск всех частых наборов и генерация устойчивых ассоциативных правил на основе найденных частых наборов [4].

Набор, имеющий поддержку не ниже $minsup$, называют *частым*, иначе набор называют *редким*. *Поддержкой набора* $I \subseteq \mathcal{I}$ является доля транзакций $\mathcal{D}$, содержащих данный набор:

$$support(I) = \frac{|\{T \in \mathcal{D} \mid I \subseteq T.I\}|}{|\mathcal{D}|}. \tag{2}$$

Множество всех частых k-наборов обозначают $\mathcal{L}_k$. Решением задачи *поиска частых наборов* будет множество $\mathcal{L} = \cup_{k=1}^{k_{max}} \mathcal{L}_k$, где k_{max} — максимальное количество объектов в частом наборе.

Классическим алгоритмом решения задачи поиска частых наборов является алгоритм *Apriori* [4]. Идея Apriori заключается в итеративной генерации множества *кандидатов* в частые наборы и последующем отборе кандидатов с подходящим значением поддержки. Итерация осуществляется по k, мощности наборов-кандидатов, начиная с 1. В алгоритме используется следующее свойство *антимонотонности поддержки* (принцип *a priori*), которое позволяет отсеивать заведомо редкие наборы: если k-набор является редким, то содержащий его $(k+1)$-набор также является редким. Узким местом Apriori является операция генерации и проверки наборов-кандидатов, поскольку при достаточно больших значениях k и малых значениях $minsup$ имеют место значительные накладные расходы на поддержку наборов-кандидатов и повторяющиеся операции сканирования множества транзакций и подсчета поддержки. Разработан ряд улучшений алгоритма Apriori, связанных с сокращением количества наборов-кандидатов, количества просматриваемых транзакций и количества операций сканирования: алгоритмы *AprioriTid* [4], *DHP* [65], *Partition* [70], *DIC* [14], *Eclat* [81] и др.

Альтернативой подходу с использованием кандидатов для решения задачи поиска частых наборов является алгоритм *FP-Growth* [32]. На первой фазе FP-Growth за две операции полного сканирования множества транзакций выполняется построение специальной структуры данных, *FP-дерева (FP tree, frequent pattern tree)*, которое в компактном виде хранит наборы и их поддержку. На второй фазе с помощью рекурсивного обхода построенного дерева осуществляется генерация частых наборов. FP-Growth предъявляет большие требования к объему необходимой оперативной памяти.

Улучшением данного подхода являются алгоритмы *AFOPT* [43], *OpportuneProject* [44] и др. В настоящее время отсутствует алгоритм поиска частых наборов, превосходящий все остальные для всех возможных вариаций множества транзакций и порогового значения поддержки *minsup* [34].

1.2. Задача кластеризации

Задача *кластеризации (clustering)* заключается в разбиении множества объектов сходной структуры на заранее неизвестные группы (кластеры) в зависимости от похожести свойств объектов. Кластеризация применяется в широком спектре приложений: сегментирование медицинских и спутниковых изображений, анализ ДНК-микрочипов и текстов и др.

Формальное определение задачи кластеризации выглядит следующим образом. Пусть заданы конечные множества: $X = \{x_1, x_2, \ldots, x_n\}$, где $n > 1$ — множество объектов d-мерного метрического пространства, для которых задана функция расстояния $\rho(x_i, x_j)$, и $C = \{c_1, c_2, \ldots, c_k\}$, где $k \ll n$ — набор уникальных идентификаторов (номеров, имен, меток) кластеров.

Алгоритм (четкой) кластеризации определяется как функция $\alpha : X \to C$, которая каждому объекту назначает уникальный идентификатор кластера. Алгоритм кластеризации выполняет разбиение множества X на непересекающиеся непустые подмножества (*кластеры*) таким образом, чтобы каждый кластер состоял из объектов, близких по метрике ρ, а объекты разных кластеров существенно отличались. *Алгоритм нечеткой кластеризации* позволяет одному и тому же объекту принадлежать одновременно всем кластерам, но с различной степенью принадлежности.

Алгоритм *разделительной (partitioning) кластеризации* предполагает начальное разбиение исходного множества объектов на кластеры (возможно, выполняемое случайным образом), при котором в каждом кластере имеется, по крайней мере, один объект, и каждый объект принадлежит в точности одному кластеру. Далее итеративно осуществляется перемещение объектов между кластерами с целью улучшить начальное разбиение (чтобы объекты из одного кластера были более «близкими», а из разных кластеров — более «далекими» друг другу).

В алгоритме *k-Means* [46] при улучшении разбиения каждый кластер представляется посредством среднего значения координат объектов в кластере. Для представления кластеров в разделительных алгоритмах могут использоваться также медиана или мода координат объектов (алгоритмы *k-Median* [28] и *k-Mode* [36] соответственно).

Алгоритмы *k-Medoids* и *PAM (Partitioning Around Medoids)* [39] в качестве представления каждого кластера используют тот объект подвергаемого кластеризации множества, который находится ближе остальных к центру кластера. Техника медоидов направлена на повышение устойчивости алгоритма к выбросам и шумам в данных (робастности).

Иерархическая кластеризация заключается в последовательном разбиении исходного множества объектов по уровням иерархии. *Агломеративный иерархический алгоритм* начинает работу в предположении, что каждый исходный объект образует отдельный кластер, и затем выполняет слияние близких друг к другу объектов или кластеров до тех пор, пока не будет получен единственный кластер или не будет выполнено условие завершения слияния. Примером агломеративного подхода является алгоритм *AGNES* [39].

Дивизимный иерархический алгоритм, напротив, стартует, предполагая, что все исходные объекты входят в один кластер, и затем итеративно выполняет его разбиение на менее мощные кластеры до тех пор, пока не будут получены кластеры-синглтоны или не будет выполнено условие завершения слияния. Дивизимный подход реализован в алгоритме *DIANA* [39].

Плотностная (density-based) кластеризация предполагает добавление объектов (называемых в контексте плотностных методов точками) в кластер до тех пор, пока плотность (количество) соседних точек не превысит некоторого наперед заданного значения порога концентрации. Плотностная кластеризация используется для нахождения аномалий и кластеров произвольной формы (в отличие от разделительных алгоритмов, которые приспособлены для нахождения кластеров сферической формы). Типичным представителем плотностной кластеризации является алгоритм *DBSCAN* [23], осуществляющий построение кластера как множества точек близкой плотности, которое имеет наибольшую мощность.

Решеточная (grid-based) кластеризация предполагает разбиение пространства исходных данных на конечное число ячеек, формирующих решеточную структуру, над которой выполняются операции, необходимые для кластеризации. Алгоритм *STING* [84] использует статистическую информацию, хранящуюся в прямоугольных ячейках решетки. Статистические данные о ячейках верхних уровней вычисляются на основе статистических данных о ячейках нижних уровней. Для кластеризации используются следующие статистические данные: количество точек в ячейке, минимальное, максимальное, среднее значение атрибутов и др.

2. Подходы к интеграции анализа данных в СУБД

Исследования в области интеграции интеллектуального анализа данных в реляционные системы баз данных начаты в конце XX в., практически одновременно с зарождением интеллектуального анализа данных как самостоятельной научной дисциплины. В работах Агравала (Agrawal) и Сараваджи (Sarawagi) [5, 68], где предложен термин «связывание» (coupling) интеллектуального анализа данных и СУБД. Хан (Han) предложил [30] различать следующие виды интеграции интеллектуального анализа данных в СУБД: слабое связывание, среднее связывание и сильное связывание.

При *слабом связывании (loose coupling)* система интеллектуального анализа данных отделена от СУБД и использует сервисы СУБД для экспорта исходных данных из хранилища и импорта результатов анализа обратно в хранилище данных. Данный подход использует большинство современных открытых систем для интеллектуального анализа данных: KNIME [9], Weka [24] и др.

При *среднем связывании (semitight coupling)* система интеллектуального анализа данных также отделена от СУБД, но применяет СУБД для реализации некоторых примитивных операций, часто используемых при подготовке данных для интеллектуального анализа. В качестве таких операций могут фигурировать индексирование, соединение отношений, построение гистограмм, статистические вычисления (поиск максимума и минимума, стандартного отклонения) и др. Помимо этого СУБД может обеспечивать хранение предварительно вычисленных и часто используемых промежуточных результатов интеллектуального анализа.

При *сильном связывании (tight coupling)* система интеллектуального анализа данных рассматривается как функциональная единица СУБД, которая обеспечивает выполнение запросов пользователя на анализ данных в базе данных, подобно тому как машина баз данных исполняет запросы SQL в приложениях OLTP (оперативной обработки транзакций). В этом случае функции интеллектуального анализа данных реализуются и оптимизируются на основе использования структур данных, схем индексирования и методов обработки запросов, встроенных в СУБД. Сильное связывание предпочтительно с точки зрения удобства прикладного программиста и конечного пользователя, но одновременно является наиболее трудоемким в реализации [30].

В рамках исследования подходов к реализации сильного связывания можно выделить следующие два основных направления работ: исследование методов создания систем анализа данных в СУБД и разработка методов реализации алгоритмов интеллектуального анализа данных на SQL.

Система интеллектуального анализа данных может быть реализована как *внедренная в СУБД подсистема*, которая поддерживает специальный язык аналитических запросов или расширяет SQL соответствующими конструкциями. Машина баз данных при этом модифицируется, чтобы осуществлять разбор, оптимизацию и выполнение запроса.

Реализация системы интеллектуального анализа данных возможна также в виде *медиатора (посредника)* между прикладным программистом баз данных и СУБД. Прикладному программисту предоставляется графический интерфейс или специализированный язык для формирования запросов интеллектуального анализа данных. Медиатор преобразует запросы интеллектуального анализа данных в набор запросов на SQL и/или вызовов хранимых процедур, которые затем исполняет СУБД.

Альтернативой для системного программиста, реализующего внедрение анализа данных в СУБД, является разработка *библиотеки хранимых процедур. Хранимая процедура (stored procedure)* представляет собой текст подпрограммы, компилируемый однократно и постоянно хранимый на сервере базы данных. Хранимая процедура похожа на подпрограммы языков высокого уровня (имеет параметры, локальные переменные и др.), но может возвращать результат запроса SQL. Такая подпрограмма может быть реализована на SQL или его процедурном расширении, либо на языке высокого уровня (эта возможность, как правило, поддерживается в современных СУБД). Подключая библиотеку к своему приложению базы данных, прикладной программист получает возможность выполнять интеллектуальный анализ данных, не выходя за рамки СУБД.

Реализация на SQL алгоритмов интеллектуального анализа данных предполагает, что исходные и промежуточные данные алгоритма, а также результаты его работы будут представлены в виде реляционных таблиц. Обработка указанных таблиц реализуется посредством запросов SQL, что обеспечивает потенциальную переносимость алгоритмов на другие реляционные СУБД. Однако, поскольку SQL является декларативным языком запросов, разработка в нем алгоритмов анализа данных сопряжена с определенными трудностями. Например, в SQL затруднена реализация структур данных в оперативной памяти (список, бинарное дерево, граф и др.) и агрегации данных по столбцам таблицы (штатные функции SQL `COUNT`, `MIN`, `MAX`, `AVG` выполняют только построчную агрегацию).

Одним из основных путей преодоления подобных проблем является разработка пользовательских функций, расширяющих штатные возможности SQL. *Пользовательская функция (user-defined function, UDF)* представляет собой хранимую на сервере баз данных

подпрограмму-функцию, вызов которой может быть включен в качестве выражения в оператор SQL, а результат вычисляется в рамках выполнения соответствующего запроса. Пользовательская функция допускает результат как скалярного, так и табличного типа. Реализация пользовательской функции может быть выполнена на SQL, процедурном расширении SQL либо на языке высокого уровня. Например, в работе [58] описан подход к реализации пользовательских функций, выполняющих агрегатные операции по столбцам реляционных таблиц.

Помимо расширения функциональности SQL, пользовательские функции могут в общем случае более эффективно, чем штатные средства СУБД, реализовать операции агрегации, математический вычисления и др. (за счет возможности уменьшить количество операций сканирования таблиц, переноса части вычислений в оперативную память и др.) [58, 59, 62]. Например, в работе [62] предложена реализация метода главных компонент в параллельной СУБД на основе пользовательских функций, использующих библиотеку параллельных подпрограмм Intel Math Kernel Library; в работе [60] предложен способ ускорения вычисления Байесовской модели для линейной регрессии на основе использования параллельных пользовательских функций. Следует также отметить, что обратной стороной подобного увеличения эффективности пользовательских функций является возможная потеря переносимости алгоритмов анализа данных в другие СУБД.

3. Методы разработки систем анализа данных в СУБД

В данном разделе рассмотрены заметные научные исследования в области разработки систем и библиотек анализа данных в СУБД, а также расширений языка баз данных SQL.

3.1. Системы и библиотеки анализа данных

```
select
  PredictAssociation([HealthMiningModel].[AssocLines], INCLUDE_STATISTICS, 3)
  from [HealthMiningModel]
    natural prediction join (
    select
      60 as [Age],
      TRUE as [isSmoker],
      'Pneumonia' as [Disease]) as [AssocLines]
```

Рис. 1. Пример запроса на языке DMX

В корпорации Microsoft разработаны стандарт OLE DB for Data Mining и язык запросов *DMX (Data Mining Extensions)* [78], используемые в ее продукте MS SQL Server Analysis Services. Стандарт специфицирует интерфейс программирования приложений (Application Programming Interface, API) интеллектуального анализа данных. Язык DMX имеет SQL-подобный синтаксис (операторы определения и манипулирования данными и др.), однако его операндами являются не реляционные отношения, а модели интеллектуального анализа данных. Под моделью интеллектуального анализа данных понимается сочетание самих данных, алгоритма интеллектуального анализа данных и коллекции значений параметров и фильтров, управляющих использованием и обработкой данных. Пример запроса на языке DMX показан на рис. 1.

```
DBMS_DATA_MINING.CREATE_MODEL (
  model_name           => 'credit_risk_model',
  function             => DBMS_DATA_MINING.classification,
  data_table_name      => 'credit_card_data',
  case_id_column_name => 'customer_id',
  target_column_name   => 'credit_risk',
  settings_table_name => 'credit_risk_model_settings');

select customer_name
from credit_card_data
where PREDICTION (credit_risk_model using *) = 'LOW'
  and customer_value = 'HIGH'
```

Рис. 2. Пример запроса на языке Oracle Data Mining

Подобный подход реализован также в коммерческой СУБД Oracle в виде модуля Oracle Data Mining [77]. На рис. 2 приведен пример создания модели классификации и запроса к ней.

Ванг (Wang) и др. разработали систему интеллектуального анализа данных *ATLAS* [83], которая поддерживает одноименный язык запросов, являющийся надстройкой над SQL. Язык ATLAS добавляет в SQL поддержку пользовательских функций и функций, возвращающих в качестве значения реляционную таблицу. На языке ATLAS реализованы алгоритм поиска шаблонов Apriori, алгоритм кластеризации DBSCAN [23] и классификация посредством деревьев решений.

```
-- Кластеризация k средних
kmeanspp(
  rel_source, -- имя таблицы с входными данными
  expr_point, -- имя колонки с данными
  k, -- количество искомых кластеров
  fn_dist, -- вид функции расстояния
  agg_centroid, -- вид агрегационной функции при расчете центроидов
  max_num_iterations, -- максимальное количество итераций
  min_frac_reassigned, -- минимальное количество переназначаемых объектов
  для останова вычислений
  seeding_sample_ratio) -- размер сэмпла данных для инициализации центроидов

select * from madlib.kmeanspp(
 'km_sample', 'points', 2, 'madlib.squared_dist_norm2',
 'madlib.avg', 20, 0.001);
```

Рис. 3. Пример функции библиотеки MADlib

Хеллерштейн (Hellerstein) и др. разработали библиотеку *MADlib* [33] с открытым исходным кодом для интеллектуального анализа данных в реляционных СУБД PostgreSQL и Greenplum. MADlib предоставляет богатый набор алгоритмов анализа данных (кластеризация, классификация, регрессия и др.), адаптированные для использования в реляционной СУБД и не требующие экспорта и импорта данных внешних аналитических приложений. В реализации MADlib используются пользовательские функции, написанные разработчиками на язык программирования Python, которые обеспечивают обращения к словарю базы данных и формирование корректной структуры таблиц с выходными

данными для заданных таблиц с входными данными. Пример интерфейса и вызова функции библиотеки MADlib приведен на рис. 3.

```
-- Объекты
create table Items (
  item integer primary key, -- Уникальный ИД объекта
  description varchar)     -- Описание объекта

-- Транзакции
create table D (
  tid integer,  -- Уникальный ИД транзакции
  item integer, -- Уникальный ИД объекта в транзакции
  pimary key (tid,item),
  foreign key item references Items (item))

-- Наборы
create table Sets (
  sid integer,  -- Уникальный ИД набора
  item integer, -- Уникальный ИД объекта в наборе
  pimary key (sid,item),
  foreign key item references Items (item))

-- Поддержка наборов
create table Support (
  sid integer pimary key, -- Уникальный ИД набора
  supp real)         -- Поддержка набора

-- Ассоциативные правила
create table Rules (
  rid integer,  -- Уникальный ИД правила
  sida integer, -- Уникальный ИД набора-антецедента
  sidc integer, -- Уникальный ИД набора-консеквента
  sid integer,  -- Уникальный ИД набора-объединения антецедента и консеквента
  supp, conf real -- Поддержка и достоверность правила
  foreign key sida references Sets (sid),
  foreign key sidc references Sets (sid),
  foreign key sid references Sets (sid))
```

Рис. 4. База данных для виртуальных представлений поиска шаблонов

В цикле работ [11–13] Блокилом (Blockeel), Гоэталсом (Goethals) и др. предложена система интеллектуального анализа данных в СУБД на основе виртуальных аналитических представлений. *Виртуальное аналитическое представление (virtual mining view)* создается как именованный запрос к таблицам базы данных и другим представлениям, который обеспечивает логическое хранение (в отличие от физического хранения таблиц базы данных) результатов интеллектуального анализа данных. При выполнении запроса пользователя к такому представлению в СУБД срабатывает системный триггер, который запускает алгоритм интеллектуального анализа данных. Далее СУБД материализует кортежи, запрошенные пользователем. Система поддерживает построение виртуальных аналитических представлений для поиска шаблонов и классификации с помощью деревьев решений. На рис. 4 приведена схема базы данных, используемой для построения виртуальных аналитических представлений поиска шаблонов. Данное исследование сконцентрировано, однако, на частичной или полной материализации запрошенных

пользователем результатов анализа и не затрагивает вопрос исполнения аналитических алгоритмов внутри СУБД.

В работе [61] Ордонезом и др. предложена облачная система интеллектуального анализа данных на основе реляционной СУБД. На локальной машине запускается реляционная СУБД, подключающаяся к облаку. База данных хранится и обрабатывается в облаке, а в локальную СУБД передаются только результаты анализа. Помимо возможностей обработки данных только на локальной машине или только в облаке, система поддерживает режим гибридного исполнения, когда выполняется распределение вычислительной нагрузки между облаком и локальной СУБД.

Ордонезом и др. также разработана система интеллектуального анализа данных, основанный на использовании реляционной СУБД и хранимых процедур [59]. Технологический цикл работы с системой выглядит следующим образом. Анализируемые данные, параметры аналитических алгоритмов и проч. хранится в в реляционных таблицах. Клиентское приложение соединяется с сервером СУБД по протоколу ODBC. С помощью графического интерфейса конечный пользователь специфицирует задачу интеллектуального анализа данных, ее параметры и таблицы исходных данных. Приложение запускает хранимую процедуру, которая, в свою очередь, выполняет генерацию необходимых SQL запросов. Вычислительно трудоемкие операции (например, вычисления, связанные с матрицами) выполняются с помощью предварительно созданных и откомпилированных соответствующих пользовательских функций. Графический интерфейс позволяет выполнять мониторинг выполнения аналитического алгоритма (время, количество итераций и др.) и последующую визуализацию результатов.

Махаян (Mahajan) и др. разработали систему анализа DAnA [47], которая выполняет автоматическое преобразование запросов на выполнение анализа данных в исходный код для выполнения на реконфигурируемых вычислительных системах FPGA. Реализация данного преобразования выполняется с помощью пользовательской функции на SQL, использующей язык Python. Система DAnA предполагает интеграцию в СУБД на основе специализированных аппаратных устройств, называемых *страйдерами (striders)*. Страйдер имеет прямой интерфейс доступа к буферному пулу СУБД и выполняет извлечение, очистку и обработку кортежей данных, которые затем передаются на ускоритель FPGA для параллельного исполнения аналитического алгоритма. Использование FPGA позволяет ускорить вычисления ценой, однако, потери переносимости разработанного решения, поскольку требует включения в состав системы специализированных аппаратных устройств (страйдеров).

В работе [2] Речкалов описал систему поиска шаблонов, реализованную в СУБД на основе предложенного языка XML-разметки алгоритмов поиска частых наборов, реализуемых на SQL. Разработана разметка алгоритмов ScanOnce [82] и SETM [35]. Используя разметку, система выполняет автоматическую генерацию хранимых процедур на языке SQL в зависимости от специфицированных пользователем таблиц исходных данных и параметров алгоритма. Среди полученных SQL реализаций система выбирает для исполнения наиболее эффективный, используя имеющуюся в составе современных СУБД команду `EXPLAIN`, которая позволяет получить стоимость (относительную оценку времени исполнения) запроса SQL без его фактического выполнения.

3.2. Языки запросов и расширения SQL для интеллектуального анализа данных

```
find association rules as HealthRuleSet
related to Salary, Age, isSmoker, Disease
from HealthDB
where Disease='Pneumonia' and Age>60
with support threshold=0.05
with confidence threshold=0.07
```

Рис. 5. Пример запроса на языке *DMQL*

Одним из первых языков интеллектуального анализа данных можно является язык *DMQL* [29], предложенный в 1996 г. Хан (Han) и др. DMQL предоставляет SQL-подобный синтаксис для записи запросов интеллектуального анализа данных. Примитивы DMQL позволяют определить данные, подлежащие анализу, решаемую задачу интеллектуального анализа (классификация, поиск ассоциативных правил и др.), семантические иерархии в анализируемых данных и пороговые значения параметров задачи (поддержка и др.). Пример запроса на языке *DMQL* приведен на рис. 5. Язык запросов DMQL был реализован в рамках системы анализа данных в СУБД DBMiner [29].

Язык DMQL позднее послужил основой для разработки целого ряда языков запросов интеллектуального анализа данных: язык для анализа временных данных *TQML* [20] Чена (X. Chen), 1998 г.; языки для анализа географических данных *GMQL* [31] Хана (Han), 1997 г. и *SDMQL* [49] Малербы (Malerba), 2004 г.; язык для анализа пространственно-временных данных *ST-DMQL* [15] Богорны (Bogorny), 2009 г.

```
mine rule HealthRuleSet as
  select distinct 1..n Disease as body,
  1..1 isSmoker as head
  from HealthDB
  where body.Disease='Pneumonia' and body.Age>60
  extracting rules with
  support: 0.1
  confidence: 0.3
```

Рис. 6. Пример запроса на языке MINE RULE

Meo (Meo) и др. в 1996 г. предложили SQL-подобный оператор *MINE RULE* [51], который предназначен для решения задачи поиска ассоциативных правил. Пример запроса с использованием оператора *MINE RULE* показан на рис. 6.

Позднее в 1999 г. в работе [37] Имилински (Imielinski) описал язык *MSQL*, представляющий собой расширение SQL для решения задачи поиска ассоциативных правил. В отличие от DMQL, язык MSQL предполагает не только нахождение ассоциативных правил, но и предоставляет возможность выборки результирующих правил. Соответствующие примеры запросов приведены на рис. 7.

Следует отметить, что описанные выше расширения языка баз данных не позволяют явно манипулировать полученными результатами интеллектуального анализа данных (подобно тому, как это обеспечивается в SQL). В этом смысле интересной является

```
GetRules(HealthDB)
  into HealthRuleSet R
  where R.Body in {(Disease=*), (Age=*), (Salary=*)}
  and R.Body has {(Disease='Pneumonia'), (Age>60)}
  and R.Consequent in {(isSmoker=*)}
  and Support>0.1
  and Confidence>0.7

SelectRules(HealthRuleSet)
  where Body has {(Disease='Pneumonia')}
  and {(Salary>0) and (Salary<=1000)}
  and Support>0.1
  and Confidence>0.7
```

Рис. 7. Пример запроса на языке MSQL

работа [17] Калдерса (Calders) и др., в которой предложены специализированные модель базы данных и алгебра для интеллектуального анализа данных.

```
select ST_Area(ST_Polygon(House.location))
from House
where House.household_income < 30000
cluster by House.location
```

Рис. 8. Пример запроса с оператором `CLUSTER BY`

Сан (Sun) и др. в работе [76] предложили расширить язык SQL оператором *`CLUSTER BY`* для кластеризации данных. Данная конструкция подразумевает выполнение группировки строк результата запроса в соответствии со специфицированным алгоритмом кластеризации, в отличие от стандартного оператора `GROUP BY`, который осуществляет группировку по точному совпадению значений в полях записей. На рис. 8 приведен пример использования указанного оператора в запросе, который задействует PostGIS [45], расширение СУБД PosgreSQL, обеспечивающее поддержку географических объектов. Силва (Silva) и др. в работе [73] предложили схожий по назначению оператор *`SIMILAR GROUP BY`*, реализованный авторами в СУБД PostgreSQL.

4. Реализация алгоритмов анализа данных на SQL

Реализация алгоритмов интеллектуального анализа данных в виде набора запросов SQL обеспечивает переносимость между различными СУБД. Далее приведен обзор работ, описывающих применение данного подхода для решения различных задач интеллектуального анализа данных.

4.1. Задача поиска шаблонов

Одной из первых SQL-реализаций задачи поиска частых наборов является алгоритм *SETM* [35], разработанный Хутсмой (Houtsma) и др. в 1995 г. В основе алгоритма лежит использование запросов SQL, выполняющих поиск частых наборов без использования принципа *a priori*, приведенных на рис. 9 (здесь и далее используются обозначения из раздела 1.1, определение таблицы D приведено на рис. 4). В 2000 г. Йосизава (Yoshizawa) и др. работе [85] предложили улучшение данного алгоритма, предполагающее использование

```
insert into L_k
  select d_1.item, ..., d_k.item, count(*)
  from L_{k-1} ℓ, D d_1, ..., D d_k
  where d_1.tid= ... = d_k.tid and
    d_1.item = ℓ.item_1 and
    ...
    d_{k-l}.item = ℓ.item_{k-1} and
    d_k.item > d_{k-1}.item
  group by d_1.item, ..., d_k.item
  having count(*) >= :minsup
```

Рис. 9. Запросы SQL для поиска частых наборов в алгоритме SETM [35]

представлений вместо некоторых таблиц, и рефакторинг запросов на основе использования подзапросов.

```
insert into C_k
  select ℓ_1.item_1, ..., ℓ_1.item_{k-1}, ℓ_2.item_{k-1}
  from L_{k-1} ℓ_1, L_{k-1} ℓ_2
  where ℓ_1.item_1=ℓ_2.item_1 and
    ...
    ℓ_1.item_{k-2}=ℓ_2.item_{k-2} and
    ℓ_1.item_{k-1}<ℓ_2.item_{k-1}
```

Рис. 10. Запросы SQL для генерации наборов-кандидатов в алгоритмах из работы [68]

В 1998 г. Сараваджи (Sarawagi) и др. [68] предложили алгоритмы *K-Way-Join*, *Three-Way-Join*, *Subquery* и *Two-Group-Bys*, которые основаны на классическом алгоритме Apriori [4] для оперативной памяти. SQL-реализация генерации наборов-кандидатов в указанных алгоритмах представлена на рис. 10.

```
insert into L_k
  select C_k.item_1, ..., C_k.item_k, count(*)
  from C_k, D d_1, ..., D d_k
  where d_1.item_1=C_k.item_1 and
    ...
    d_k.item_k=C_k.item_k and
    d_1.tid=d_2.tid and
    ...
    d_{k-1}.tid=d_k.tid
  group by C_k.item_1, ..., C_k.item_k
  having count(*) >= :minsup
```

Рис. 11. Запросы SQL для поиска частых наборов в алгоритме K-Way-Join [68]

Алгоритмы отличаются способом вычисления поддержки наборов-кандидатов. На рис. 11 представлен способ вычисления поддержки в алгоритме K-Way-Join. В алгоритме Three-Way-Join (см. рис. 12) для снижения количества затратных операций естественного соединения используется следующая модификация. В дополнение к полям (item_1, ..., item_k) в таблицу C_k добавляются три новых поля: (oid, id_1, id_2), где oid — уникальный идентификатор набора, а id_1 и id_2 — уникальные идентификаторы тех наборов из $\mathcal{L}_{k-1}$, которые использованы при создании данного набора. Кроме того, на k-м просмотре

базы транзакций создается дополнительная таблица T_k с полями (tid, oid), которая для каждого идентификатора транзакции tid хранит каждый идентификатор oid набора из C_k, входящего в транзакцию.

```
-- Создание дополнительной таблицы
insert into Tk
  select t1.tid, oid
  from Ck, Tk-1 t1, Tk-1 t2
  where t1.oid=Ck.id1 and t2.oid=Ck.id2 and t1.tid=t2.tid

-- Подсчет поддержки с помощью дополнительной таблицы
insert into Lk
  select Ck.oid, Ck.item1, ..., Ck.itemk, cnt
  from Ck, (
    select oid as cid, count(*) as cnt
    from Ck
    group by oid
    having count(*) >= :minsup)
  where Ck.oid=cid
```

Рис. 12. Запросы SQL для поиска частых наборов в алгоритме Three-Way-Join [68]

Томас (Thomas) и Чакраварти (Chakravarthy) в 1999 г. предложили алгоритм *Set-oriented Apriori* [79], в основе которого лежит идея сокращения вычислений при подсчете поддержки наборов за счет построения на каждом шаге подсчета дополнительной таблицы для хранения транзакций, содержащих соответствующее количество объектов. Эксперименты показали, что алгоритм Set-oriented Apriori показывает лучшую производительность, чем алгоритм Subquery.

Ранцау (Rantzau) в 2004 г. разработал алгоритм *Quiver* [66] для решения задачи поиска частых наборов, основанный на предложенной им с коллегами реляционной операции универсального квантования (universal quantification) [67]. Данная операция является аналогом операции деления в реляционной алгебре. Эксперименты на разработанном авторами исполнителе запросов, который поддерживает новую операцию, показывают, что алгоритм Quiver способен показать лучшую производительность, чем другие алгоритмы, реализованные на SQL. В то же время при выполнении на существующих коммерческих СУБД алгоритм Quiver заметно проигрывает в производительности другим алгоритмам, реализованным на SQL, поскольку в данных СУБД отсутствует эффективная реализация предложенной авторами операции.

В работе [82] Ванг (Wang) и др. описали алгоритм *ScanOnce* на языке PL/SQL СУБД Oracle, использующий курсоры. *Курсор* представляет собой указатель на область памяти, в которой хранится результат выполнения запроса, и позволяет осуществлять последовательное сканирование этого результата. В данном алгоритме организуется цикл по уникальным идентификаторам наборов, на каждом шаге которого создается курсор, указывающий на объекты, входящие в соответствующий набор. Далее осуществляется последовательное сканирование курсора и подсчет поддержки набора.

Алашкур (Alashqur) в 2010 г. разработал алгоритм *RDB-MINER* [7] поиска ассоциативных правил в реляционных таблицах. Алгоритм основан на классическом алгоритме Apriori и использовании динамически формируемых запросов SQL.

Алгоритм *Propad* [71], предложенный Шангом (Shang) и др. в 2004 г., представляет собой реализацию классического алгоритма FP-Growth [32] на SQL. Подобно классическому алгоритму, в данном алгоритме не выполняется затратная операция генерации наборов-кандидатов. Алгоритм Propad демонстрирует лучшую производительность, чем алгоритм K-Way-Join (подобно тому, как классический алгоритм FP-Growth является более производительным, чем классический алгоритм Apriori [4]). Позднее Сидло (Sidló) и Лукаш (Lukács) разработали алгоритм *FP-TDG* [72], который также представляет собой SQL-реализацию классического алгоритма FP-Growth [32].

4.2. Задача кластеризации

Ордонезом (Ordonez) в работах [55, 56] на языке SQL реализован разделительный алгоритм кластеризации k-Means [46]. Им же в сотрудничестве с коллегами в работах [48, 57] выполнена реализация на SQL алгоритма кластеризации EM [26]. В работе [1] Миниахметов и др. представили реализацию алгоритма нечеткой кластеризации данных Fuzzy C-Means [10] в СУБД PostgreSQL.

В указанных работах используется следующая техника индексного представления матричных данных. Пусть требуется организовать хранение матрицы кластеризуемых объектов $X \in \mathbb{R}^{n\times d}$. Естественным способом хранения будет таблица с заголовком $(x_1, x_2, \ldots, x_d)$, где каждое поле x_i имеет вещественный тип. Однако в этом случае отсутствует возможность применения агрегатных функций SQL для вычислений, выполняемых по столбцам таблицы (например, подсчет функции ρ расстояния между объектами), поскольку указанные функции осуществляют агрегацию только по строкам. Поэтому матрица X представляется в виде таблицы из $n{\cdot}d$ записей, которая имеет заголовок $(\underline{i,\ell}, val)$, где поля i и ℓ имеют целочисленный тип, а val — вещественный. В составном первичном ключе (i, ℓ) таблицы поле i указывает номер исходного объекта, поле ℓ — номер координаты этого объекта; поле val таблицы хранит значения координат объектов.

В работе [42] Лепиниоти (Lepinioti) разработал алгоритм иерархической кластеризации *Cobweb/IDX*. Реализация выполнена на языке PL/SQL для СУБД Oracle. Данный алгоритм является инкрементальным (поддерживает кластеризацию по мере появления новых данных).

В работе [64] Пан и др. представили подход к кластеризации вершин графа с помощью параллельной СУБД. Спроектирована реляционная база данных для хранения исходных и промежуточных данных алгоритма. Граф представляется в виде реляционной таблицы со списком ребер. Таблицы базы данных подвергаются горизонтальной фрагментации, полученные фрагменты распределяются по узлам вычислительного кластера. Каждый фрагмент обрабатывается отдельно экземпляром параллельной СУБД для получения таблицы вершин графа с метками кластеров. Обработка выполняется с помощью запросов SQL, реализующих стадии огрубления и восстановления графа в соответствии с алгоритмом кластеризации графа в оперативной памяти Кариписа—Кумара [38].

4.3. Другие задачи интеллектуального анализа данных

Помимо рассмотренных выше задач поиска шаблонов и кластеризации, SQL также используется для решения задачи классификации. Классификация похожа на задачу кластеризации в том, что ставит своей целью распределение по группам (классам) конечного числа объектов, имеющих сходную структуру в зависимости от похожести

их свойств. Отличие заключается в том, что в задаче классификации количество и семантика классов известны заранее. Одним из основных подходов к классификации является построение *дерева решений (decision tree)* [16]. Дерево решений представляет собой ориентированное дерево, в котором каждой внутренней вершине соответствует операция проверки значения указанного атрибута классифицируемых объектов, каждая дуга соответствует переходу к другой вершине в соответствии с результатом проверки, а каждому листу соответствует один из классов. Классификация на основе деревьев решений реализована в работах Саттлера и др. [69] и Ковальского и др. [41], а также Моертини (Moertini) и др. [52] для объектно-реляционных СУБД.

Несмотря на «нереляционную» природу графовых данных, внедрение интеллектуального анализа графов в реляционные СУБД является одним из актуальных направлений исследований. Падманабхан (Padmanabhan) и др. в работе [63] предложили подход к анализу структуры графов на основе использования SQL. Срихари (Srihari) и др. в работе [74] описали подход к поиску полного подграфа неориентированного графа, основанный на применении реляционной СУБД. Алгоритмы поиска часто встречающихся подграфов в графе, ориентированные на использование SQL, предложены Чакраварти (Chakravarthy) и др. и Ордонезом (Ordonez) и др. в работах [18] и [27] соответственно. Исследования, направленные на поиск циклов в графе с помощью реляционной СУБД, описаны Балачандраном (Balachandran) и др. в работе [8]. МакКафри (McCaffrey) разработал комбинированный подход к разбиению графов, использующий встраивание запросов SQL в реализацию алгоритма обработки графа на языке программирования высокого уровня [50].

Заключение

В настоящее время реляционные СУБД являются основным инструментом управления базами данных, несмотря на появление большого количества NoSQL СУБД. Одним из перспективных направлений развития реляционных СУБД является внедрение в них средств интеллектуального анализа данных. Интеллектуальный анализ данных направлен на извлечение доступных для понимания знаний, необходимых для принятия решений в различных сферах человеческой деятельности. Интеграция позволяет как избежать накладных расходов по экспорту анализируемых данных из хранилища и импорту результатов анализа обратно в хранилище, так и использовать при анализе данных системные сервисы, заложенные в архитектуре СУБД.

В статье представлен обзор методов и подходов к решению задачи интеграции интеллектуального анализа данных в реляционные СУБД. Наиболее предпочтительным с точки зрения удобства прикладного программиста и конечного пользователя является подход, предполагающий сильное связывание СУБД и технологий интеллектуального анализа данных. В этом случае система интеллектуального анализа данных рассматривается как функциональная единица СУБД, которая обеспечивает прозрачное выполнение запросов пользователя на анализ данных, хранимых в базе данных (подобно тому как машина баз данных исполняет запросы SQL в приложениях оперативной обработки транзакций). Функции интеллектуального анализа данных реализуются и оптимизируются на основе использования структур данных, схем индексирования и методов обработки запросов, встроенных в СУБД.

Рассмотрены системы интеллектуального анализа данных, которые реализуются как внедряемые в СУБД подсистемы, поддерживающие специальный язык запросов анализа данных или расширяющие SQL соответствующими конструкциями. Машина баз данных такой СУБД модифицируется, чтобы осуществлять разбор, оптимизацию и выполнение запросов анализа данных. Представлены расширения языка баз данных SQL, обеспечивающие синтаксическую поддержку интеллектуального анализа данных в СУБД.

Представлены системы интеллектуального анализа данных, реализуемые в виде медиатора (посредника) между прикладным программистом баз данных и СУБД. Прикладному программисту предоставляется графический интерфейс или специализированный язык для формирования запросов анализа данных. Медиатор преобразует запросы анализа данных в набор запросов на SQL и/или вызовов хранимых процедур, которые затем исполняет СУБД.

Приведены примеры внедрения анализа данных в СУБД на основе разработки библиотеки хранимых процедур. Хранимая процедура представляет собой текст подпрограммы, компилируемый однократно и постоянно хранимый на сервере базы данных. Хранимая процедура похожа на подпрограммы языков высокого уровня, но может возвращать результат запроса SQL. Подключая библиотеку к своему приложению базы данных, прикладной программист получает возможность выполнять интеллектуальный анализ данных, не выходя за рамки СУБД.

Дан краткий обзор основных основных задач интеллектуального анализа данных и алгоритмов их решения, рассмотрены примеры реализации указанных алгоритмов на SQL.

Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (грант № 17-07-00463), Правительства РФ в соответствии с Постановлением № 211 от 16.03.2013 (соглашение № 02.A03.21.0011) и Министерства образования и науки РФ (государственное задание 2.7905.2017/8.9).

Литература

1. Миниахметов Р.М., Цымблер М.Л. Интеграция алгоритма кластеризации Fuzzy c-Means в PostgreSQL // Вычислительные методы и программирование: Новые вычислительные технологии. 2012. Т. 13. С. 46–52.

2. Речкалов Т.В. Подход к интеграции интеллектуального анализа данных в реляционную СУБД на основе генерации текстов хранимых процедур // Вестник Южно-Уральского государственного университета. Серия: Вычислительная математика и информатика. 2013. Т. 2, № 1. С. 114–121.

3. Agrawal R., Ailamaki A., Bernstein P.A. et al. The Claremont Report on Database Research // Commun. ACM. 2009. Vol. 52, No. 6. P. 56–65. DOI: 10.1145/1516046.1516062.

4. Agrawal R., Srikant R. Fast Algorithms for Mining Association Rules in Large Databases // VLDB'94, Proceedings of 20th International Conference on Very Large Data Bases, September 12–15, 1994, Santiago de Chile, Chile. 1994. P. 487–499.

5. Agrawal R., Shim K. Developing Tightly-coupled Data Mining Applications on a Relational Database System // Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA. 1996. P. 287–290.

6. Abadi D., Agrawal R., Ailamaki A. et al. The Beckman Report on Database Research // Commun. ACM. 2016. Vol. 59, No. 2. P. 92–99. DOI: 10.1145/2845915.

7. Alashqur A. RDB-MINER: A SQL-Based Algorithm for Mining True Relational Databases // Journal of Software. 2010. Vol. 5, No. 9. P. 998–1005. DOI: 10.4304/jsw.5.9.998-1005.

8. Balachandran R., Padmanabhan S., Chakravarthy S. Enhanced DBSubdue: Supporting Subtle Aspects of Graph Mining Using a Relational Approach // Advances in Knowledge Discovery and Data Mining, 10th Pacific-Asia Conference, PAKDD 2006, Singapore, April 9–12, 2006, Proceedings. 2006. P. 673–678. DOI: 10.1007/11731139_77.

9. Berthold M.R., Cebron N., Dill F. et al. KNIME - the Konstanz Information Miner: Version 2.0 and Beyond // SIGKDD Explorations. 2009. Vol. 11, No. 1. P. 26–31. DOI: 10.1145/1656274.1656280.

10. Bezdek J.C., Ehrlich R., Full W. FCM: The Fuzzy C-Means Clustering Algorithm // Computers and Geosciences. 1984. Vol. 10, No. 2. P. 191–203. DOI: 10.1016/0098-3004(84)90020-7.

11. Blockeel H., Calders T., Fromont E. et al. An Inductive Database Prototype Based on Virtual Mining Views // Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24–27, 2008. 2008. P. 1061–1064. DOI: 10.1145/1401890.1402019.

12. Blockeel H., Calders T., Fromont E. et al. An Inductive Database Prototype Based on Virtual Mining Views // Data Min. Knowl. Discov. 2012. Vol. 24, No. 1. P. 247–287. DOI: 10.1007/s10618-011-0229-7.

13. Blockeel H., Calders T., Fromont E. et al. Inductive Querying with Virtual Mining Views // Inductive Databases and Constraint-Based Data Mining. Ed. by S. Dzeroski, B. Goethals, P. Panov. Springer, 2010. P. 265–287. DOI: 10.1007/978-1-4419-7738-0_11.

14. Brin S., Motwani R., Ullman J.D., Tsur S. Dynamic Itemset Counting and Implication Rules for Market Basket Data // SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data, May 13–15, 1997, Tucson, Arizona, USA. 1997. P. 255–264. DOI: 10.1145/253260.253325.

15. Bogorny V., Kuijpers B., Alvares L.O. ST-DMQL: A Semantic Trajectory Data Mining Query Language // International Journal of Geographical Information Science. 2009. Vol. 23, No. 10. P. 1245–1276.

16. Breiman L., Friedman J., Olshen R., Stone C. Classification and Regression Trees. Wadsworth International Group, 1984.

17. Calders T., Lakshmanan L.V.S., Ng R.T., Paredaens J. Expressive Power of an Algebra for Data Mining // ACM Trans. Database Syst. 2006. Vol. 31, No. 4. P. 1169–1214. DOI: 10.1145/1189769.1189770.

18. Chakravarthy S., Pradhan S. DB-FSG: An SQL-based Approach for Frequent Subgraph Mining // Database and Expert Systems Applications, 19th International Conference, DEXA 2008, Turin, Italy, September 1–5, 2008. Proceedings. 2008. P. 684–692. DOI: 10.1007/978-3-540-85654-2_59.

19. Chaudhuri S. What Next?: a Half-dozen Data Management Research Goals for Big Data and the Cloud // Proceedings of the 31st ACM SIGMODSIGACT- SIGART Symposium on

Principles of Database Systems, PODS 2012, Scottsdale, AZ, USA, May 20–24, 2012. 2012. P. 1–4. DOI: 10.1145/2213556.2213558.

20. Chen X., Petrounias I. Language Support for Temporal Data Mining // Principles of Data Mining and Knowledge Discovery, 2nd European Symposium, PKDD '98, Nantes, France, September 23–26, 1998, Proceedings. 1998. P. 282–290. DOI: 10.1007/BFb0094830.

21. Codd E.F. A Relational Model of Data for Large Shared Data Banks // Commun. ACM. 1970. Vol. 13, No. 6. P. 377–387. DOI: 10.1145/362384.362685.

22. Davoudian A., Chen L., Liu M. A Survey on NoSQL Stores // ACM Comput. Surv. 2018. Vol. 51, No. 2. P. 40:1–40:43. DOI: 10.1145/3158661.

23. Ester M., Kriegel H., Sander J., Xu X. A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise // Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA. 1996. P. 226–231.

24. Frank E., Hall M.A., Holmes G. et al. WEKA - A Machine Learning Workbench for Data Mining // The Data Mining and Knowledge Discovery Handbook. / Ed. by O. Maimon, L. Rokach. Springer, 2005. P. 1305–1314.

25. Frawley W.J., Piatetsky-Shapiro G., Matheus C.J. Knowledge Discovery in Databases: an Overview // Knowledge Discovery in Databases. AAAI/MIT Press, 1991. P. 1–30.

26. Dempster A., Laird N., Rubin D. Maximum Likelihood Estimation from Incomplete Data via the EM Algorithm // Journal of The Royal Statistical Society. 1977. Vol. 39, No. 1. P. 1–38.

27. Garcia W., Ordonez C., Zhao K., Chen P. Efficient Algorithms Based on Relational Queries to Mine Frequent Graphs // Proceedings of the 3rd PhD Workshop on Information and Knowledge Management, PIKM 2010, Toronto, Ontario, Canada, October 30, 2010. 2010. P. 17–24. DOI: 10.1145/1871902.1871906.

28. Guha S., Mishra N., Motwani R., O'Callaghan L. Clustering Data Streams // Proceedings of the 41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12–14 November 2000, Redondo Beach, California, USA. 2000. P. 359–366. DOI: 10.1109/SFCS.2000.892124.

29. Han J., Fu Y., Wang W. et al. DBMiner: A System for Mining Knowledge in Large Relational Databases // Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA. 1996. P. 250–255.

30. Han J., Kamber M. Data Mining: Concepts and Techniques. Morgan Kaufmann, 2006. P. 743.

31. Han J., Koperski K., Stefanovic N. GeoMiner: A System Prototype for Spatial Data Mining // SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data, May 13–15, 1997, Tucson, Arizona, USA. 1997. P. 553–556. DOI: 10.1145/253260.253404.

32. Han J., Pei J., Yin Y. Mining Frequent Patterns without Candidate Generation // Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 16–18, 2000, Dallas, Texas, USA. 2000. P. 1–12. DOI: 10.1145/342009.335372.

33. Hellerstein J.M., Re C., Schoppmann F. et al. The MADlib Analytics Library or MAD Skills, the SQL // PVLDB. 2012. Vol. 5, No. 12. P. 1700–1711.

34. HooshSadat M., Samuel H.W., Patel S., Zaiane O.R. Fastest Association Rule Mining Algorithm Predictor (FARM-AP) // Proceedings of the 4th International C* Conference on Computer Science and Software Engineering, C3S2E 2011, Montreal, Quebec, Canada, May 16–18, 2011. P. 43–50. DOI: 10.1145/1992896.1992902.

35. Houtsma M.A.W., Swami A.N. Set-Oriented Mining for Association Rules in Relational Databases // Proceedings of the 11th International Conference on Data Engineering, March 6–10, 1995, Taipei, Taiwan. 1995. P. 25–33. DOI: 10.1109/ICDE.1995.380413.

36. Huang Z. Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Values // Data Min. Knowl. Discov. 1998. Vol. 2, No. 3. P. 283–304. DOI: 10.1023/A:1009769707641.

37. Imielinski T., Virmani A. MSQL: A Query Language for Database Mining // Data Min. Knowl. Discov. 1999. Vol. 3, No. 4. P. 373–408. DOI: 10.1023/A:1009816913055.

38. Karypis G., Kumar V. Analysis of Multilevel Graph Partitioning // Proceedings of Supercomputing '95, San Diego, CA, USA, December 4–8, 1995. 1995. P. 29. DOI: 10.1145/224170.224229.

39. Kaufman L., Rousseeuw P.J. Finding Groups in Data: an Introduction to Cluster Analysis. John Wiley, 1990. DOI: 10.1002/9780470316801.

40. Krause C., Johannsen D., Deeb R. et al. An SQL-Based Query Language and Engine for Graph Pattern Matching // Graph Transformation - 9th International Conference, ICGT 2016, in Memory of Hartmut Ehrig, Held as Part of STAF 2016, Vienna, Austria, July 5–6, 2016, Proceedings. 2016. P. 153–169. DOI: 10.1007/978-3-319-40530-8_10.

41. Kowalski M., Stawicki S. SQL-based Heuristics for Selected KDD Tasks over Large Data Sets // Proceedings of the FedCSIS 2012, Federated Conference on Computer Science and Information Systems, Wroclaw, Poland, 9–12 September 2012. IEEE, 2012. P. 303–310.

42. Lepinioti K., McKearney S. Integrating Cobweb with a Relational Database // Proceedings of the International MultiConference of Engineers and Computer Scientists 2007, IMECS 2007, March 21–23, 2007, Hong Kong, China. 2007. P. 868–873.

43. Liu G., Lu H., Lou W. et al. Efficient Mining of Frequent Patterns Using Ascending Frequency Ordered Prefix-Tree // Data Min. Knowl. Discov. 2004. Vol. 9, No. 3. P. 249–274. DOI: 10.1023/B:DAMI.0000041128.59011.53.

44. Liu J., Pan Y., Wang K., Han J. Mining Frequent Item Sets by Opportunistic Projection // Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, July 23–26, 2002, Edmonton, Alberta, Canada. 2002. P. 229–238. DOI: 10.1145/775047.775081.

45. Lizardo E.O., Davis C.A. A PostGIS Extension to Support Advanced Spatial Data Types and Integrity Constraints // Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS 2017, Redondo Beach, CA, USA, November 7–10, 2017. P. 33:1–33:10. DOI: 10.1145/3139958.3140020.

46. Lloyd S.P. Least Squares Quantization in PCM // IEEE Transactions on Information Theory. 1982. Vol. 28, No. 2. P. 129–136. DOI: 10.1109/TIT.1982.1056489.

47. Mahajan D., Kim J.K., Sacks J. et al. In-RDBMS Hardware Acceleration of Advanced Analytics // PVLDB. 2018. Vol. 11, No. 11. P. 1317–1331.

48. Matusevich D.S., Ordonez C. A Clustering Algorithm Merging MCMC and EM Methods Using SQL Queries // Proceedings of the 3rd International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications, BigMine 2014, New York City, USA, August 24, 2014. 2014. P. 61–76.

49. Malerba D., Appice A., Ceci M. A Data Mining Query Language for Knowledge Discovery in a Geographical Information System // Database Support for Data Mining Applications: Discovering Knowledge with Inductive Queries. 2004. P. 95–116. DOI: 10.1007/978-3-540-44497-8_5.

50. McCaffrey J.D. A Hybrid System for Analyzing Very Large Graphs // Ninth International Conference on Information Technology: New Generations, ITNG 2012, Las Vegas, Nevada, USA, April 16–18, 2012. 2012. P. 253–257. DOI: 10.1109/ITNG.2012.43.

51. Meo R., Psaila G., Ceri S. A New SQL-like Operator for Mining Association Rules // VLDB'96, Proceedings of 22th International Conference on Very Large Data Bases, September 3–6, 1996, Mumbai (Bombay), India. 1996. P. 122–133.

52. Moertini V., Sitohang B., Santosa O.S. Searching Object-Relational

53. Ordonez C. Statistical Model Computation with UDFs // IEEE Trans. Knowl. Data Eng. 2010. Vol. 22, No. 12. P. 1752–1765. DOI: 10.1109/TKDE.2010.44.

54. Ordonez C. Can We Analyze Big Data Inside a DBMS? // Proceedings of the 16th International Workshop on Data Warehousing and OLAP, DOLAP 2013, San Francisco, CA, USA, October 28, 2013. 2013. P. 85–92. DOI: 10.1145/2513190.2513198.

55. Ordonez C. Programming the K-means Clustering Algorithm in SQL // Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22–25, 2004. 2004. P. 823–828. DOI:10.1145/1014052.1016921.

56. Ordonez C. Integrating K-Means Clustering with a Relational DBMS Using SQL // IEEE Trans. Knowl. Data Eng. 2006. Vol. 18, No. 2. P. 188–201. DOI: 10.1109/TKDE.2006.31.

57. Ordonez C., Cereghini P. SQLEM: Fast Clustering in SQL Using the EM Algorithm // Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 16–18, 2000, Dallas, Texas, USA. 2000. P. 559–570. DOI: 10.1145/342009.335468.

58. Ordonez C., Chen Z. Horizontal Aggregations in SQL to Prepare Data Sets for Data Mining Analysis // IEEE Trans. Knowl. Data Eng. 2012. Vol. 24, No. 4. P. 678–691. DOI: 10.1109/TKDE.2011.16.

59. Ordonez C., Garcia-Alvarado C. A Data Mining System Based on SQL Queries and UDFs for Relational Databases // Proceedings of the 20th ACM Conference on Information and Knowledge Management, CIKM 2011, Glasgow, United Kingdom, October 24–28, 2011. 2011. P. 2521–2524. DOI: 10.1145/2063576.2064008.

60. Ordonez C., Garcia-Alvarado C., Baladandayuthapani V. Bayesian Variable Selection in Linear Regression in One Pass for Large Datasets // TKDD. 2014. Vol. 9, No. 1. P. 3:1–3:14. DOI: 10.1145/2629617.

61. Ordonez C., Garcia-Garcia J., Garcia-Alvarado C. et al. Data Mining Algorithms as a Service in the Cloud Exploiting Relational Database Systems // Proceedings of the ACM SIGMOD

International Conference on Management of Data, SIGMOD 2013, New York, NY, USA, June 22–27, 2013. 2013. P. 1001–1004. DOI: 10.1145/2463676.2465240.

62. Ordonez C., Mohanam N., Garcia-Alvarado C. PCA for Large Data Sets with Parallel Data Summarization // Distributed and Parallel Databases. 2014. Vol. 32, No. 3. P. 377–403. DOI: 10.1007/s10619-013-7134-6.

63. Padmanabhan S., Chakravarthy S. HDB-Subdue: A Scalable Approach to Graph Mining // Data Warehousing and Knowledge Discovery, 11th International Conference, DaWaK 2009, Linz, Austria, August 31 – September 2, 2009, Proceedings. 2009. P. 325–338. DOI: 10.1007/978-3-642-03730-6_26.

64. Pan C., Zymbler M. Very Large Graph Partitioning by Means of Parallel DBMS // Advances in Databases and Information Systems - 17th East European Conference, ADBIS 2013, Genoa, Italy, September 1–4, 2013. Proceedings. 2013. P. 388–399. DOI: 10.1007/978-3-642-40683-6_29.

65. Park J.S., Chen M., Yu P.S. An Effective Hash Based Algorithm for Mining Association Rules // Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data, San Jose, California, May 22–25, 1995. 1995. P. 175–186. DOI: doi.org/10.1145/223784.223813.

66. Rantzau R. Frequent Itemset Discovery with SQL Using Universal Quantification // Database Support for Data Mining Applications: Discovering Knowledge with Inductive Queries. 2004. P. 194–213. DOI: 10.1007/978-3-540-44497-8_10.

67. Rantzau R., Shapiro L.D., Mitschang B., Wang Q. Algorithms and Applications for Universal Quantification in Relational Databases // Information Systems. 2003. Vol. 28, No. 1–2. P. 3–32. DOI: 10.1016/S0306-4379(02)00047-9.

68. Sarawagi S., Thomas S., Agrawal R. Integrating Mining with Relational Database Systems: Alternatives and Implications // SIGMOD 1998, Proceedings ACM SIGMOD International Conference on Management of Data, June 2–4, 1998, Seattle, Washington, USA. 1998. P. 343–354. DOI: 10.1145/276304.276335.

69. Sattler K.-U., Dunemann O. SQL Database Primitives for Decision Tree Classifiers // Proceedings of the 2001 ACM CIKM International Conference on Information and Knowledge Management, Atlanta, Georgia, USA, November 5–10, 2001. ACM, 2001. P. 379–386. DOI: 10.1145/502585.502650.

70. Savasere A., Omiecinski E., Navathe S.B. An Efficient Algorithm for Mining Association Rules in Large Databases // VLDB'95, Proceedings of 21th International Conference on Very Large Data Bases, September 11–15, 1995, Zurich, Switzerland. 1995. P. 432–444.

71. Shang X., Sattler K., Geist I. SQL Based Frequent Pattern Mining with FP-Growth // Applications of Declarative Programming and Knowledge Management, 15th International Conference on Applications of Declarative Programming and Knowledge Management, INAP 2004, and 18th Workshop on Logic Programming, WLP 2004, Potsdam, Germany, March 4–6, 2004, Revised Selected Papers. 2004. P. 32–46. DOI: 10.1007/11415763_3.

72. Sidlo C.I., Lukacs A. Shaping SQL-based Frequent Pattern Mining Algorithms // Knowledge Discovery in Inductive Databases, 4th International Workshop, KDID 2005, Porto, Portugal, October 3, 2005, Revised Selected and Invited Papers. 2005. P. 188–201. DOI: 10.1007/11733492_11.

73. Silva Y.N., Aref W.G., Ali M.H. Similarity Group-By // Proceedings of the 25th International Conference on Data Engineering, ICDE 2009, March 29, 2009 – April 2, 2009, Shanghai, China. 2009. P. 904–915. DOI: 10.1109/ICDE.2009.113.

74. Srihari S., Chandrashekar S., Parthasarathy S. A Framework for SQLBased Mining of Large Graphs on Relational Databases // Advances in Knowledge Discovery and Data Mining, 14th Pacific-Asia Conference, PAKDD 2010, Hyderabad, India, June 21–24, 2010. Proceedings. Part II. 2010. P. 160–167. DOI: 10.1007/978-3-642-13672-6_16.

75. Stonebraker M., Madden S., Dubey P. Intel "Big Data" Science and Technology Center Vision and Execution Plan // SIGMOD Record. 2013. Vol. 42, No. 1. P. 44–49. DOI: 10.1145/2481528.2481537.

76. Sun P., Huang Y., Zhang C. Cluster-By: An Efficient Clustering Operator in Emergency Management Database Systems // Web-Age Information Management - WAIM 2013 International Workshops: HardBD, MDSP, BigEM, TMSN, LQPM, BDMS, Beidaihe, China, June 14–16, 2013. Proceedings. 2013. P. 152–164. DOI: 10.1007/978-3-642-39527-7_17.

77. Tamayo P., Berger C., Campos M.M., et al. Oracle Data Mining - Data Mining in the Database Environment // The Data Mining and Knowledge Discovery Handbook. Ed. by O. Maimon, L. Rokach. Springer, 2005. P. 1315–1329.

78. Tang Z., Maclennan J., Kim P.P. Building Data Mining Solutions with OLE DB for DM and XML for analysis // SIGMOD Record. 2005. Vol. 34, No. 2. P. 80–85. DOI: 10.1145/1083784.1083805.

79. Thomas S., Chakravarthy S. Performance Evaluation and Optimization of Join Queries for Association Rule Mining // Data Warehousing and Knowledge Discovery, 1st International Conference, DaWaK'99, Florence, Italy, August 30 – September 1, 1999, Proceedings. 1999. P. 241–250. DOI: 10.1007/3-540-48298-9_26.

80. Turner V., Gantz J., Reinsel D., et al. The Digital Universe of Opportunities: Rich Data and the Increasing Value of the Internet of Things. 2014. URL: http://www.emc.com/leadership/digital-universe/2014iview/executive-summary.htm (дата обращения: 05.02.2019).

81. Zaki M.J. Scalable Algorithms for Association Mining // IEEE Trans. Knowl. Data Eng. 2000. Vol. 12, No. 3. P. 372–390. DOI: 10.1109/69.846291.

82. Wang F., Gordon J., Helian N. SQL Implementation of a ScanOnce Algorithm for Large Database Mining // Engineering Federated Information Systems, Proceedings of the 5th Workshop EFIS 2003, July 17–18 2003, Coventry, UK. 2003. P. 43–45.

83. Wang H., Zaniolo C., Luo C. ATLAS: A Small but Complete SQL Extension for Data Mining and Data Streams // VLDB. 2003. P. 1113–1116.

84. Wang W., Yang J., Muntz R.R. STING: A Statistical Information Grid Approach to Spatial Data Mining // VLDB'97, Proceedings of 23rd International Conference on Very Large Data Bases, August 25–29, 1997, Athens, Greece. 1997. P. 186–195.

85. Yoshizawa T., Pramudiono I., Kitsuregawa M. SQL Based Association Rule Mining Using Commercial RDBMS (IBM DB2 UBD EEE) // Data Warehousing and Knowledge Discovery, Second International Conference, DaWaK 2000, London, UK, September 4–6, 2000, Proceedings. 2000. P. 301–306. DOI: 10.1007/3-540-44466-1_30.

Цымблер Михаил Леонидович, к.ф.-м.н., доцент, кафедра системного программирования, Южно-Уральский государственный университет (национальный исследовательский университет) (Челябинск, Российская Федерация)

DOI: 10.14529/cmse190203

OVERVIEW OF METHODS FOR INTEGRATING DATA MINING INTO DBMS

South Ural State University (pr. Lenina 76, Chelyabinsk, 454080 Russia)
E-mail: mzym@susu.ru

Received: 27.02.2019

Data Mining is aimed to discovering understandable knowledge from data, which can be used for decision-making in various fields of human activity. The Big Data phenomenon is a characteristic feature of the modern information society. The processes of cleaning and structuring Big data lead to the formation of very large databases and data warehouses. Despite the emergence of a large number of NoSQL DBMSs, the main database management tool is still relational DBMS. Integration of Data Mining into relational DBMS is one of the promising directions of development of relational databases. Integration allows both to avoid the overhead of exporting the analyzed data from the repository and importing the analysis results back to the repository, as well as using system services embedded in the DBMS architecture for data analysis. The paper provides an overview of methods and approaches to solving the problem of integrating data mining in a DBMS. A classification of approaches to solving the problem of integrating data mining in a DBMS is given. The SQL database language extensions to provide syntactic support for data mining in a DBMS are introduced. Examples of the implementation of data mining algorithms for SQL and data analysis systems in relational databases are considered.

Keywords: data mining, relational DBMS, classification, clustering, pattern mining.

FOR CITATION

Zymbler M.L. Overview of Methods for Integrating Data Mining into DBMS. *Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering.* 2019. vol. 8, no. 2. pp. 32–62. (in Russian) DOI: 10.14529/cmse190203.

References

1. Miniakhmetov R.M., Zymbler M.L. Integration of Fuzzy c-Means Clustering algorithm with PostgreSQL database management system. *Vychislitel'nye Metody i Programmirovanie* [Numerical Methods and Programming]. 2012. vol. 13. pp. 46–52.

2. Rechkalov T.V. An Approach to Integration of Data Mining with Relational DBMS Based on Automatic SQL Code Generation. *Vestnik Yuzho-Uralskogo Gosudarstvennogo Universiteta. Seriya "Vychislitel'naya Matematika i Informatika"* [Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering]. 2013. vol. 2, no. 1. pp. 114–121.

3. Agrawal R., Ailamaki A., Bernstein P.A. et al. The Claremont Report on Database Research. *Commun. ACM.* 2009. vol. 52, no. 6. pp. 56–65. DOI: 10.1145/1516046.1516062.

4. Agrawal R., Srikant R. Fast Algorithms for Mining Association Rules in Large Databases. VLDB'94, Proceedings of 20th International Conference on Very Large Data Bases, September 12–15, 1994, Santiago de Chile, Chile. 1994. pp. 487–499.

5. Agrawal R., Shim K. Developing Tightly-coupled Data Mining Applications on a Relational Database System. Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA. 1996. pp. 287–290.

6. Abadi D., Agrawal R., Ailamaki A. et al. The Beckman Report on Database Research. *Commun. ACM*. 2016. vol. 59, no. 2. pp. 92–99. DOI: 10.1145/2845915.

7. Alashqur A. RDB-MINER: A SQL-Based Algorithm for Mining True Relational Databases. *Journal of Software*. 2010. vol. 5, no. 9. pp. 998–1005. DOI: 10.4304/jsw.5.9.998-1005.

8. Balachandran R., Padmanabhan S., Chakravarthy S. Enhanced DBSubdue: Supporting Subtle Aspects of Graph Mining Using a Relational Approach. Advances in Knowledge Discovery and Data Mining, 10th Pacific-Asia Conference, PAKDD 2006, Singapore, April 9–12, 2006, Proceedings. 2006. pp. 673–678. DOI: 10.1007/11731139_77.

9. Berthold M.R., Cebron N., Dill F. et al. KNIME - the Konstanz Information Miner: Version 2.0 and Beyond. *SIGKDD Explorations*. 2009. vol. 11, no. 1. pp. 26–31. DOI: 10.1145/1656274.1656280.

10. Bezdek J.C., Ehrlich R., Full W. FCM: The Fuzzy C-Means Clustering Algorithm. *Computers and Geosciences*. 1984. vol. 10, no. 2. pp. 191–203. DOI: 10.1016/0098-3004(84)90020-7.

11. Blockeel H., Calders T., Fromont E. et al. An Inductive Database Prototype Based on Virtual Mining Views. Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24–27, 2008. 2008. pp. 1061–1064. DOI: 10.1145/1401890.1402019.

12. Blockeel H., Calders T., Fromont E. et al. An Inductive Database Prototype Based on Virtual Mining Views. *Data Min. Knowl. Discov.* 2012. vol. 24, no. 1. pp. 247–287. DOI: 10.1007/s10618-011-0229-7.

13. Blockeel H., Calders T., Fromont E. et al. Inductive Querying with Virtual Mining Views. Inductive Databases and Constraint-Based Data Mining. Ed. by S. Dzeroski, B. Goethals, P. Panov. Springer, 2010. pp. 265–287. DOI: 10.1007/978-1-4419-7738-0_11.

14. Brin S., Motwani R., Ullman J.D., Tsur S. Dynamic Itemset Counting and Implication Rules for Market Basket Data. SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data, May 13–15, 1997, Tucson, Arizona, USA. 1997. pp. 255–264. DOI: 10.1145/253260.253325.

15. Bogorny V., Kuijpers B., Alvares L.O. ST-DMQL: A Semantic Trajectory Data Mining Query Language. *International Journal of Geographical Information Science*. 2009. vol. 23, no. 10. pp. 1245–1276.

16. Breiman L., Friedman J., Olshen R., Stone C. Classification and Regression Trees. Wadsworth International Group, 1984.

17. Calders T., Lakshmanan L.V.S., Ng R.T., Paredaens J. Expressive Power of an Algebra for Data Mining. *ACM Trans. Database Syst.* 2006. vol. 31, no. 4. pp. 1169–1214. DOI: 10.1145/1189769.1189770.

18. Chakravarthy S., Pradhan S. DB-FSG: An SQL-based Approach for Frequent Subgraph Mining. Database and Expert Systems Applications, 19th International Conference, DEXA 2008, Turin, Italy, September 1–5, 2008. Proceedings. 2008. pp. 684–692. DOI: 10.1007/978-3-540-85654-2_59.

19. Chaudhuri S. What Next?: a Half-dozen Data Management Research Goals for Big Data and the Cloud. Proceedings of the 31st ACM SIGMODSIGACT- SIGART Symposium on Principles of Database Systems, PODS 2012, Scottsdale, AZ, USA, May 20–24, 2012. 2012. pp. 1–4. DOI: 10.1145/2213556.2213558.

20. Chen X., Petrounias I. Language Support for Temporal Data Mining. Principles of Data Mining and Knowledge Discovery, 2nd European Symposium, PKDD '98, Nantes, France, September 23–26, 1998, Proceedings. 1998. pp. 282–290. DOI: 10.1007/BFb0094830.

21. Codd E.F. A Relational Model of Data for Large Shared Data Banks. *Commun. ACM*. 1970. vol. 13, no. 6. pp. 377–387. DOI: 10.1145/362384.362685.

22. Davoudian A., Chen L., Liu M. A Survey on NoSQL Stores. ACM Comput. Surv. 2018. vol. 51, no. 2. pp. 40:1–40:43. DOI: 10.1145/3158661.

23. Ester M., Kriegel H., Sander J., Xu X. A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA. 1996. pp. 226–231.

24. Frank E., Hall M.A., Holmes G. et al. WEKA - A Machine Learning Workbench for Data Mining. The Data Mining and Knowledge Discovery Handbook. / Ed. by O. Maimon, L. Rokach. Springer, 2005. pp. 1305–1314.

25. Frawley W.J., Piatetsky-Shapiro G., Matheus C.J. Knowledge Discovery in Databases: an Overview. Knowledge Discovery in Databases. AAAI/MIT Press, 1991. pp. 1–30.

26. Dempster A., Laird N., Rubin D. Maximum Likelihood Estimation from Incomplete Data via the EM Algorithm. *Journal of The Royal Statistical Society*. 1977. vol. 39, no. 1. pp. 1–38.

27. Garcia W., Ordonez C., Zhao K., Chen P. Efficient Algorithms Based on Relational Queries to Mine Frequent Graphs. Proceedings of the 3rd PhD Workshop on Information and Knowledge Management, PIKM 2010, Toronto, Ontario, Canada, October 30, 2010. 2010. pp. 17–24. DOI: 10.1145/1871902.1871906.

28. Guha S., Mishra N., Motwani R., O'Callaghan L. Clustering Data Streams. Proceedings of the 41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12–14 November 2000, Redondo Beach, California, USA. 2000. pp. 359–366. DOI: 10.1109/SFCS.2000.892124.

29. Han J., Fu Y., Wang W. et al. DBMiner: A System for Mining Knowledge in Large Relational Databases. Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA. 1996. pp. 250–255.

30. Han J., Kamber M. Data Mining: Concepts and Techniques. Morgan Kaufmann, 2006. pp. 743.

31. Han J., Koperski K., Stefanovic N. GeoMiner: A System Prototype for Spatial Data Mining. SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data, May 13–15, 1997, Tucson, Arizona, USA. 1997. pp. 553–556. DOI: 10.1145/253260.253404.

32. Han J., Pei J., Yin Y. Mining Frequent Patterns without Candidate Generation. Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 16–18, 2000, Dallas, Texas, USA. 2000. pp. 1–12. DOI: 10.1145/342009.335372.

33. Hellerstein J.M., Re C., Schoppmann F. et al. The MADlib Analytics Library or MAD Skills, the SQL. *PVLDB*. 2012. vol. 5, no. 12. pp. 1700–1711.

34. HooshSadat M., Samuel H.W., Patel S., Zaiane O.R. Fastest Association Rule Mining Algorithm Predictor (FARM-AP). Proceedings of the 4th International C* Conference on Computer Science and Software Engineering, C3S2E 2011, Montreal, Quebec, Canada, May 16–18, 2011. pp. 43–50. DOI: 10.1145/1992896.1992902.

35. Houtsma M.A.W., Swami A.N. Set-Oriented Mining for Association Rules in Relational Databases. Proceedings of the 11th International Conference on Data Engineering, March 6–10, 1995, Taipei, Taiwan. 1995. pp. 25–33. DOI: 10.1109/ICDE.1995.380413.

36. Huang Z. Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Values. *Data Min. Knowl. Discov.* 1998. vol. 2, no. 3. pp. 283–304. DOI: 10.1023/A:1009769707641.

37. Imielinski T., Virmani A. MSQL: A Query Language for Database Mining. *Data Min. Knowl. Discov.* 1999. vol. 3, no. 4. pp. 373–408. DOI: 10.1023/A:1009816913055.

38. Karypis G., Kumar V. Analysis of Multilevel Graph Partitioning. Proceedings of Supercomputing '95, San Diego, CA, USA, December 4–8, 1995. 1995. pp. 29. DOI: 10.1145/224170.224229.

39. Kaufman L., Rousseeuw P.J. Finding Groups in Data: an Introduction to Cluster Analysis. John Wiley, 1990. DOI: 10.1002/9780470316801.

40. Krause C., Johannsen D., Deeb R. et al. An SQL-Based Query Language and Engine for Graph Pattern Matching. Graph Transformation - 9th International Conference, ICGT 2016, in Memory of Hartmut Ehrig, Held as Part of STAF 2016, Vienna, Austria, July 5–6, 2016, Proceedings. 2016. pp. 153–169. DOI: 10.1007/978-3-319-40530-8_10.

41. Kowalski M., Stawicki S. SQL-based Heuristics for Selected KDD Tasks over Large Data Sets. Proceedings of the FedCSIS 2012, Federated Conference on Computer Science and Information Systems, Wroclaw, Poland, 9–12 September 2012. IEEE, 2012. pp. 303–310.

42. Lepinioti K., McKearney S. Integrating Cobweb with a Relational Database. Proceedings of the International MultiConference of Engineers and Computer Scientists 2007, IMECS 2007, March 21–23, 2007, Hong Kong, China. 2007. pp. 868–873.

43. Liu G., Lu H., Lou W. et al. Efficient Mining of Frequent Patterns Using Ascending Frequency Ordered Prefix-Tree. *Data Min. Knowl. Discov.* 2004. vol. 9, no. 3. pp. 249–274. DOI: 10.1023/B:DAMI.0000041128.59011.53.

44. Liu J., Pan Y., Wang K., Han J. Mining Frequent Item Sets by Opportunistic Projection. Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, July 23–26, 2002, Edmonton, Alberta, Canada. 2002. pp. 229–238. DOI: 10.1145/775047.775081.

45. Lizardo E.O., Davis C.A. A PostGIS Extension to Support Advanced Spatial Data Types and Integrity Constraints. Proceedings of the 25th ACM SIGSPATIAL International Conference

on Advances in Geographic Information Systems, GIS 2017, Redondo Beach, CA, USA, November 7–10, 2017. pp. 33:1–33:10. DOI: 10.1145/3139958.3140020.

46. Lloyd S.P. Least Squares Quantization in PCM. IEEE Transactions on Information Theory. 1982. vol. 28, no. 2. pp. 129–136. DOI: 10.1109/TIT.1982.1056489.

47. Mahajan D., Kim J.K., Sacks J. et al. In-RDBMS Hardware Acceleration of Advanced Analytics. *PVLDB*. 2018. vol. 11, no. 11. pp. 1317–1331.

48. Matusevich D.S., Ordonez C. A Clustering Algorithm Merging MCMC and EM Methods Using SQL Queries. Proceedings of the 3rd International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications, BigMine 2014, New York City, USA, August 24, 2014. 2014. pp. 61–76.

49. Malerba D., Appice A., Ceci M. A Data Mining Query Language for Knowledge Discovery in a Geographical Information System. Database Support for Data Mining Applications: Discovering Knowledge with Inductive Queries. 2004. pp. 95–116. DOI: 10.1007/978-3-540-44497-8_5.

50. McCaffrey J.D. A Hybrid System for Analyzing Very Large Graphs. Ninth International Conference on Information Technology: New Generations, ITNG 2012, Las Vegas, Nevada, USA, April 16–18, 2012. 2012. pp. 253–257. DOI: 10.1109/ITNG.2012.43.

51. Meo R., Psaila G., Ceri S. A New SQL-like Operator for Mining Association Rules. VLDB'96, Proceedings of 22th International Conference on Very Large Data Bases, September 3–6, 1996, Mumbai (Bombay), India. 1996. pp. 122–133.

52. Moertini V., Sitohang B., Santosa O.S. Searching Object-Relational DBMS Features for Improving Efficiency and Scalability of Decision Tree Algorithms. iiWAS'2006 - The 8th International Conference on Information Integration and Web-based Applications Services, December 4–6, 2006, Yogyakarta, Indonesia. 2006. pp. 323–330.

53. Ordonez C. Statistical Model Computation with UDFs. *IEEE Trans. Knowl. Data Eng.* 2010. vol. 22, no. 12. pp. 1752–1765. DOI: 10.1109/TKDE.2010.44.

54. Ordonez C. Can We Analyze Big Data Inside a DBMS?. Proceedings of the 16th International Workshop on Data Warehousing and OLAP, DOLAP 2013, San Francisco, CA, USA, October 28, 2013. 2013. pp. 85–92. DOI: 10.1145/2513190.2513198.

55. Ordonez C. Programming the K-means Clustering Algorithm in SQL. Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22–25, 2004. 2004. pp. 823–828. DOI:10.1145/1014052.1016921.

56. Ordonez C. Integrating K-Means Clustering with a Relational DBMS Using SQL. *IEEE Trans. Knowl. Data Eng.* 2006. vol. 18, no. 2. pp. 188–201. DOI: 10.1109/TKDE.2006.31.

57. Ordonez C., Cereghini P. SQLEM: Fast Clustering in SQL Using the EM Algorithm. Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 16–18, 2000, Dallas, Texas, USA. 2000. pp. 559–570. DOI: 10.1145/342009.335468.

58. Ordonez C., Chen Z. Horizontal Aggregations in SQL to Prepare Data Sets for Data Mining Analysis. *IEEE Trans. Knowl. Data Eng.* 2012. vol. 24, no. 4. pp. 678–691. DOI: 10.1109/TKDE.2011.16.

59. Ordonez C., Garcia-Alvarado C. A Data Mining System Based on SQL Queries and UDFs for Relational Databases. Proceedings of the 20th ACM Conference on Information and Knowledge Management, CIKM 2011, Glasgow, United Kingdom, October 24–28, 2011. 2011. pp. 2521–2524. DOI: 10.1145/2063576.2064008.

60. Ordonez C., Garcia-Alvarado C., Baladandayuthapani V. Bayesian Variable Selection in Linear Regression in One Pass for Large Datasets. *TKDD*. 2014. vol. 9, no. 1. pp. 3:1–3:14. DOI: 10.1145/2629617.

61. Ordonez C., Garcia-Garcia J., Garcia-Alvarado C. et al. Data Mining Algorithms as a Service in the Cloud Exploiting Relational Database Systems. Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2013, New York, NY, USA, June 22–27, 2013. 2013. pp. 1001–1004. DOI: 10.1145/2463676.2465240.

62. Ordonez C., Mohanam N., Garcia-Alvarado C. PCA for Large Data Sets with Parallel Data Summarization. *Distributed and Parallel Databases*. 2014. vol. 32, no. 3. pp. 377–403. DOI: 10.1007/s10619-013-7134-6.

63. Padmanabhan S., Chakravarthy S. HDB-Subdue: A Scalable Approach to Graph Mining. Data Warehousing and Knowledge Discovery, 11th International Conference, DaWaK 2009, Linz, Austria, August 31 – September 2, 2009, Proceedings. 2009. pp. 325–338. DOI: 10.1007/978-3-642-03730-6_26.

64. Pan C., Zymbler M. Very Large Graph Partitioning by Means of Parallel DBMS. Advances in Databases and Information Systems - 17th East European Conference, ADBIS 2013, Genoa, Italy, September 1–4, 2013. Proceedings. 2013. pp. 388–399. DOI: 10.1007/978-3-642-40683-6_29.

65. Park J.S., Chen M., Yu P.S. An Effective Hash Based Algorithm for Mining Association Rules. Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data, San Jose, California, May 22–25, 1995. 1995. pp. 175–186. DOI: doi.org/10.1145/223784.223813.

66. Rantzau R. Frequent Itemset Discovery with SQL Using Universal Quantification. Database Support for Data Mining Applications: Discovering Knowledge with Inductive Queries. 2004. pp. 194–213. DOI: 10.1007/978-3-540-44497-8_10.

67. Rantzau R., Shapiro L.D., Mitschang B., Wang Q. Algorithms and Applications for Universal Quantification in Relational Databases. *Information Systems*. 2003. vol. 28, no. 1–2. pp. 3–32. DOI: 10.1016/S0306-4379(02)00047-9.

68. Sarawagi S., Thomas S., Agrawal R. Integrating Mining with Relational Database Systems: Alternatives and Implications. SIGMOD 1998, Proceedings ACM SIGMOD International Conference on Management of Data, June 2–4, 1998, Seattle, Washington, USA. 1998. pp. 343–354. DOI: 10.1145/276304.276335.

69. Sattler K.-U., Dunemann O. SQL Database Primitives for Decision Tree Classifiers. Proceedings of the 2001 ACM CIKM International Conference on Information and Knowledge Management, Atlanta, Georgia, USA, November 5–10, 2001. ACM, 2001. pp. 379–386. DOI: 10.1145/502585.502650.

70. Savasere A., Omiecinski E., Navathe S.B. An Efficient Algorithm for Mining Association Rules in Large Databases. VLDB'95, Proceedings of 21th International Conference on Very Large Data Bases, September 11–15, 1995, Zurich, Switzerland. 1995. pp. 432–444.

71. Shang X., Sattler K., Geist I. SQL Based Frequent Pattern Mining with FP-Growth. Applications of Declarative Programming and Knowledge Management, 15th International Conference on Applications of Declarative Programming and Knowledge Management, INAP 2004, and 18th Workshop on Logic Programming, WLP 2004, Potsdam, Germany, March 4–6, 2004, Revised Selected Papers. 2004. pp. 32–46. DOI: 10.1007/11415763_3.

72. Sidlo C.I., Lukacs A. Shaping SQL-based Frequent Pattern Mining Algorithms. Knowledge Discovery in Inductive Databases, 4th International Workshop, KDID 2005, Porto, Portugal, October 3, 2005, Revised Selected and Invited Papers. 2005. pp. 188–201. DOI: 10.1007/11733492_11.

73. Silva Y.N., Aref W.G., Ali M.H. Similarity Group-By. Proceedings of the 25th International Conference on Data Engineering, ICDE 2009, March 29, 2009 – April 2, 2009, Shanghai, China. 2009. pp. 904–915. DOI: 10.1109/ICDE.2009.113.

74. Srihari S., Chandrashekar S., Parthasarathy S. A Framework for SQLBased Mining of Large Graphs on Relational Databases. Advances in Knowledge Discovery and Data Mining, 14th Pacific-Asia Conference, PAKDD 2010, Hyderabad, India, June 21–24, 2010. Proceedings. Part II. 2010. pp. 160–167. DOI: 10.1007/978-3-642-13672-6_16.

75. Stonebraker M., Madden S., Dubey P. Intel "Big Data" Science and Technology Center Vision and Execution Plan. *SIGMOD Record*. 2013. vol. 42, no. 1. pp. 44–49. DOI: 10.1145/2481528.2481537.

76. Sun P., Huang Y., Zhang C. Cluster-By: An Efficient Clustering Operator in Emergency Management Database Systems. Web-Age Information Management - WAIM 2013 International Workshops: HardBD, MDSP, BigEM, TMSN, LQPM, BDMS, Beidaihe, China, June 14–16, 2013. Proceedings. 2013. pp. 152–164. DOI: 10.1007/978-3-642-39527-7_17.

77. Tamayo P., Berger C., Campos M.M., et al. Oracle Data Mining - Data Mining in the Database Environment. The Data Mining and Knowledge Discovery Handbook. Ed. by O. Maimon, L. Rokach. Springer, 2005. pp. 1315–1329.

78. Tang Z., Maclennan J., Kim P.P. Building Data Mining Solutions with OLE DB for DM and XML for analysis. *SIGMOD Record*. 2005. vol. 34, no. 2. pp. 80–85. DOI: 10.1145/1083784.1083805.

79. Thomas S., Chakravarthy S. Performance Evaluation and Optimization of Join Queries for Association Rule Mining. Data Warehousing and Knowledge Discovery, 1st International Conference, DaWaK'99, Florence, Italy, August 30 – September 1, 1999, Proceedings. 1999. pp. 241–250. DOI: 10.1007/3-540-48298-9_26.

80. Turner V., Gantz J., Reinsel D., et al. The Digital Universe of Opportunities: Rich Data and the Increasing Value of the Internet of Things. 2014. Available at: http://www.emc.com/leadership/digital-universe/2014iview/executive-summary.htm (accessed: 05.02.2019).

81. Zaki M.J. Scalable Algorithms for Association Mining. *IEEE Trans. Knowl. Data Eng.* 2000. vol. 12, no. 3. pp. 372–390. DOI: 10.1109/69.846291.

82. Wang F., Gordon J., Helian N. SQL Implementation of a ScanOnce Algorithm for Large Database Mining. Engineering Federated Information Systems, Proceedings of the 5th Workshop EFIS 2003, July 17–18 2003, Coventry, UK. 2003. pp. 43–45.

83. Wang H., Zaniolo C., Luo C. ATLAS: A Small but Complete SQL Extension for Data Mining and Data Streams. VLDB. 2003. pp. 1113–1116.

84. Wang W., Yang J., Muntz R.R. STING: A Statistical Information Grid Approach to Spatial Data Mining. VLDB'97, Proceedings of 23rd International Conference on Very Large Data Bases, August 25–29, 1997, Athens, Greece. 1997. pp. 186–195.

85. Yoshizawa T., Pramudiono I., Kitsuregawa M. SQL Based Association Rule Mining Using Commercial RDBMS (IBM DB2 UBD EEE). Data Warehousing and Knowledge Discovery, Second International Conference, DaWaK 2000, London, UK, September 4–6, 2000, Proceedings. 2000. pp. 301–306. DOI: 10.1007/3-540-44466-1_30.