ISSN 2658 – 7505
Выпуск №3, 2019
Электронный научный журнал «Вестник молодёжной науки России»

УДК 004.658

ФОРМИРОВАНИЕ МОДЕЛИ SQL-ЗАПРОСА НА ОСНОВЕ СТАТИСТИЧЕСКИХ ПАРАМЕТРОВ ПРОИЗВОДИТЕЛЬНОСТИ

Алгазали С.М.М. *alghazali.salah@yandex.ru,* ***Айвазов В.Г.*** *olmm2@ya.ru*
Кузнецова А.В. *alvitkuz@yandex.ru*
Южно-Российский государственный политехнический университет (НПИ)
имени М.И. Платова, г. Новочеркасск

В данной статье рассматриваются вопросы формирования вектора признаков для модели SQL-запроса на основе многомерных статистических данных о результатах исполнения. Полученные модели используются для последующего кластер-анализа с целью решения проблемы эффективного поиска ресурсоёмких экземпляров запросов. Высокая размерность статистических данных, формируемых системными утилитами СУБД, обуславливает необходимость применения корреляционного и статистического анализа для выявления совокупности значимых и независимых признаков, имеющих отношение к цели исследования. Уменьшение размера модели на основе PCA-метода направленно на повышение точности кластеризации. Работа выполнена на базе статистических данных утилиты AWR СУБД Oracle.

Ключевые слова: SQL-запрос, статистические параметры производительности, информационная модель

FORMATION SQL-REQUEST MODEL ON THE BASIS OF PERFORMANCE STATISTICAL PARAMETERS

Alghazali C.M.M., Aivazov V.G., Kuznetsova A.V.
Platov South-Russian State Polytechnic University (NPI), Novocherkassk

This article discusses the formation of a vector of characteristics for SQL-queries model based on multidimensional statistical data on the results executions. The obtained models are used for subsequent cluster analysis in order to solve the problem of efficiently finding resource-intensive query instances.

The high dimensionality of statistical data generated by DBMS system utilities makes it necessary to use correlation and statistical analysis to identify a set of significant and independent features relevant to the research objective. Reducing model size based on the PCA-method aimed at increasing the accuracy of clustering.

The work is based on the statistical data of the AWR-utility Oracle DBMS.

Keywords: SQL query, statistical performance parameters, information model

Наиболее ресурсоёмкими задачами при работе крупных промышленных и корпоративных программных систем различного назначения (Business Intelligence, OnLine Transaction Processing, Data Warehousing и др.) являются задачи управления данными, то есть задачи размещения, извлечения, удаления, и редактирования информации. Для выполнения указанных манипуляций используются команды языка DML (Data Manipulation Language), являющегося подмножеством языка запросов SQL (Structured Query Language). В свою очередь, наиболее употребимой является операция извлечения информации, которая осуществляется при помощи SQL-запросов с использованием команды SELECT. В дальнейшем под SQL-запросом будем понимать прежде всего эту команду.

В рамках системного анализа SQL-запрос рассматривается как сложный объект, представленный:

1. некоторой одиночной командой к системе управления базой данных, существующей в рамках программы или программной системы, в совокупности с набором параметров, доступных для редактирования как во время разработки, так и во время выполнения программы;

2. планом исполнения, генерируемым системой управления базой данных на основании текста запроса, схемы базы данных, статистической информации по таблицам и индексам, доступных индексов, набором директив оптимизатора (hints) относительно методов доступа и методов проведения операций соединения;
3. набором реляционных операторов, соответствующих плану исполнения и непосредственно воздействующих на таблицы БД;
4. набором статистических данных о процессе исполнения запроса, которые формируются системой управления базой данных и хранится в её внутренних таблицах в течение определенного срока;
5. набором данных, возвращаемых SQL-запросом в программу.

Размещение отдельных компонентов и исполнение операторов запроса происходит в программной среде, в среде клиент-серверной СУБД и БД, которые, в свою очередь, функционируют под управлением операционных систем в рамках локальной или глобальной сети на различных аппаратных ресурсах (рисунок 1).

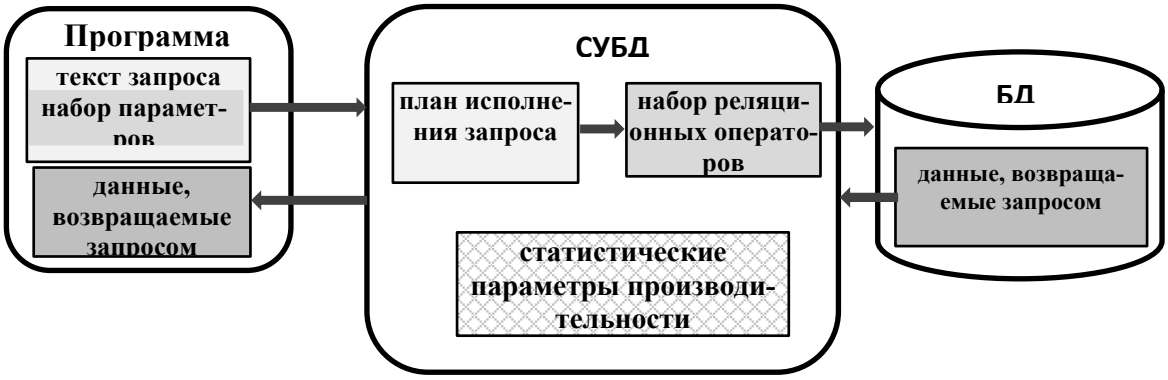

Рис. 1 – Размещение компонентов SQL-запроса

Совокупность статистических параметров производительности, записываемых сервером БД (т.е. СУБД) при выполнении SQL-выражения, характеризует его поведение во времени и отражают объем системных ресурсов, затраченных на выполнение запроса. Промышленные СУБД (Oracle, MS SQL Server, IBM DB2) аккумулируют примерно одинаковые наборы статистических параметров, число которых колеблется в пределах нескольких десятков (3-5). Способы формирования и сроки хранения статистических параметров в каждой СУБД имеют свои особенности.

Производительность SQL-запросов определяется многими факторами, одни из которых можно отнести к детерминированным (качество самого SQL-запроса, настройки и производительность системных программных и аппаратных средств, качество проектирования БД, создание и модификация индексов и др.), а другие – к случайным (присутствие/отсутствие запроса в кэше, объем и статистическое распределение обрабатываемых данных, пиковые нагрузки в сети и др.).

Несмотря на значительную долю случайной составляющей, именно данные производительности выполненных SQL-выражений служат основным источником для поиска неэффективных, ресурсоемких и длительных запросов. Поскольку количество запросов, выполняемых в единицу времени, и объем статистических данных о

результатах их исполнения для указанных выше типов программных систем очень велики, возникает необходимость в автоматическом выявлении ограниченного подмножества запросов, которые в дальнейшем будут детально исследованы на предмет их эффективности. Для этого необходимо создать комплексную информационную модель SQL-запроса, представляющую собой набор отобранных параметров, которые наилучшим образом характеризуют исследуемый объект с точки зрения производительности. Совокупность объектов, созданных на базе предложенной модели, будут подвергнуты процедуре кластеризации с целью формирования достоверннного показателя качества для каждой выделенной группы и для последующего выявления нетипичных групп и отдельных экземпляров SQL-запросов [1].

В большинстве инструментальных средств мониторинга и анализа производительности программных систем, в «собственных» и «сторонних» средствах администрирования СУБД делается упор на очевидные параметры, оценивающие производительность SQL-запросов. Совокупность параметров, каждый из которых в отдельности можно назвать малозначащим, как правило, при этом не рассматривается. Однако появившиеся и хорошо зарекомендовавшие себя технологии кластерного анализа на основе аппарата искусственных нейронных сетей, позволяющие оперировать высокоразмерными моделями, предоставляют возможность анализировать данные в их полноте. Кроме того, в наборе статистических параметров могут иметь место скрытые нелинейные и причинно-следственные зависимости, не учитываемые корреляцией. Их потенциальное наличие является дополнительным аргументом в пользу применения нейросетевой кластеризации высокоразмерных моделей объектов [2].

Из общего набора параметров к основным, характеризующим потребление ресурсов процессора и системы ввода-вывода, можно отнести: процессорное время исполнения запроса (CPU_time), общее время выполнения запроса, от получения команды до завершения (elapsed_time), количество чтений из буферного кэша (buffer_gets), количество байт, физически прочитанных с диска (physical_ read_bytes), количество операций чтения с диска (disk_read). Очевидно, что на величину этих параметров значительное влияние оказывают количество пользователей, посылающих запросы в программную систему, распределение данных в таблицах БД, состояние и настройки сети, состояние и настройки СУБД, т.е. всё то, что выше было отнесено к случайной составляющей. Использование же расширенной совокупности статистических данных произвоительности (таблица 1), по мнению авторов, даёт лучшее представление об объекте и позволяет построить более, точную модель. При этом снижается влияние случайных событий и факторов, влияющих на производительность запросов.

Представленные параметры, описывающие процесс исполнения SQL-запросов, во многих случаях взаимосвязаны, т.е. коррелируют между собой. Так, анализируя статистические данные производительности, формируемые СУБД для системы приложений электронного бизнеса Oracle E-Business Suite (OEBS), удалось определить, что сильно коррелированы между собой параметры, относящиеся к чтению данных: physical_read_bytes и disk_reads ($R \rightarrow 1{,}0$). К ним примыкают аналогичные параметры, специфичные для специализированной, высокопроизводительной подсистемы хранения данных Oracle Exadata. Параметр parse_calls сильно коррелирован с px_servers_execs в связи с тем, что при параллельном исполнении запроса для каждого

параллельного потока вызывается процедура разбора запроса. Достаточно сильно коррелированы параметры, имеющие отношение к процессам записи данных: physical_write_bytes, physical_write_requests, direct_writes; параметры времени исполнения: elapsed_time, cpu_time, apwait (R→0,8).

Таблица 1

Статистические данные производительности SQL-запросов

№	Параметр	Назначение параметра	Обоснование применимости
1.	ecxecutions	количество исполнений запроса	*важен* для оценки вклада запроса в общую нагрузку на систему
2.	sharable_mem	размер разделяемой памяти, занятой дочерним курсором (байт)	*полезен* для косвенной оценки сложности запроса
3.	optimizer_cost	суммарная оценка производительности плана исполнения	*полезен* для косвенной оценки производительности запроса
4.	loaded_versions	количество дочерниз курсоров, под которые выделена память	*полезен* для оценки используемого объема памяти.
5.	version_count	общее количество дочерних курсоров	*полезен* для оценки внешних факторов, влияющих на план исполнения
6.	physical_read_bytes	количество байт, физически прочитанных с диска	*важен* для оценки используемых ресурсов подсистемы ввода/вывода
7.	physical_write_bytes	количество байт, записанных на диск	
8.	disk_reads	количество операций физического чтения с диска	
9.	physical_read_requests	количество операций чтения с диска	
10.	physical_write_requests	количество операций записи информации на диск	
11.	rows_processed	количество обработанных и возвращенных строк	
12.	parse_calls	количество вызовов процедуры синтаксического анализа	*полезен* для косвенной оценки сложности запроса
13.	px_servers_execs	количество запусков вспомогательных процессов-серверов для параллельного исполнения запроса	*важен* для оценки производительности параллельных запросов
14.	end_of_fetch_count	количество раз, когда запрос вернул полный набор данных в случае успешного завершения	*полезен* для определения запросов, часто завершающихся с ошибкой до окончания извлечения данных
15.	fetches	количество вызовов процедуры получения строк из курсора	*полезен* для определения количества успешных выполнений
16.	buffer_gets	количество вызовов из буферного кеша (логическое чтение)	*важнен* для оценки эффективности работы буферного кеша
17.	invalidations	количество раз, когда дочерний курсор становился непригодным для использования	*полезен* для выявления большого количества операций синтаксического разбора
18.	loads	количество загрузок/ перезагрузок в память	*важен* для оценки интенсивности использования запроса
19.	sorts	количество операций сортировки	*важен* для исследования проблем нехватки памяти
20.	direct_writes	количество операций прямой записи на диск	*важен* для оценки доли прямых записей в общем количестве записей
21.	cpu_time	процессорное время для парсинга, исполнения и фетчинга для данного курсора (мкс)	*необходим* для оценки интенсивности использования CPU
22.	elapsed_time	общее время длительности запроса (мкс)	*необходим* для оценки длительности исполнения

23.	iowait	время, проведённое в ожидании пользовательского ввода-вывода (мкс)	*полезен* при учете проблем с «внешними» файловыми системами
24.	apwait	время, проведённое в ожидании приложения (мкс)	*полезен* при учете внешних факторов, связанных с работой приложения
25.	ccwait	время, проведённое в ожидании разделяемых данных и ресурсов (мкс)	*полезен* при выявлении длительных блокировок
26.	clwait	время, проведённое в ожидании завершения событий, связанных с параллелизмом (мкс)	*полезен* при выявлении проблем со связностью внутри кластера

Большое число статистических параметров, с разной степенью отажающих эффективность SQL-запроса, и наличие тесной взаимосвязи между отдельными группами одновременно наблюдаемых параметров является исходной предпосылкой для использования одного из методов факторного анализа – метода главных компонент (Principal Component Analysis – PCA). Применение этого метода позволило сократить число анализируемых параметров SQL-запроса до приемлемого уровня, и, в свою очередь, уменьшить размер и сложность последующей кластерной модели с минимальными потерями исходной информации. Предварительно параметры SQL-запросов были пронормированы на единичную выборочную дисперсию, поскольку диапазоны статистических параметров SQL-запросов имеют широкий разброс от [0..1] до [0-10^{10-13}] и различные единицы измерения.

Результаты PCA для типичного набора статистических данных, охватывающих недельный срок работы OEBS, отражены в таблице 2. Во втором столбце (Собственные значения) представлены дисперсии выделенных компонент. В третьем столбце для каждой компоненты приводится процент от общей дисперсии: первый компонент объясняет 21 процент общей дисперсии, второй компонент – 11,5 процентов, и т.д. Четвертый столбец содержит накопленную или кумулятивную дисперсию.

Таблица 2

Результаты метода главных компонент

№ компоненты	Собственные значения	Доля объясненной дисперсии, %	Куммулятивное значение объясненной дисперсии, %
1	6,819	21,31	21,31
2	3,670	11,47	32,78
3	2,673	8,35	41,13
4	2,531	7,91	49,04 (≈50%)
5	1,838	5,74	54,78 (≈50%)
6	1,660	5,19	59,97
7	1,299	4,06	64,03
8	1,112	3,48	67,51
9	1,066	3,33	70,84
10	1,031	3,22	74,06
11	1,005 (>1)	3,14	77,20
12	0,999	3,12	80,32
13	0,997	3,12	83,44
14	0,949	2,97	86,40

15	0,913	2,85	89,26
16	0,790	2,47	91,73
17	0,716	2,24	93,97
18	0,693	2,17	96,13
19	0,350	1,09	97,22
20	0,262	0,82	98,04
21	0,247	0,77	98,81
22	0,199	0,62	99,44
23	0,101	0,32	99,75
24	0,065	0,20	99,96
25	0,013	0,04	100,00
26	0,001	0,00	100,00

На рисунке 2 представлено распеределение собственных значений главных компонент, демонстрирующее значимость каждой компоненты с точки зрения её информативности.

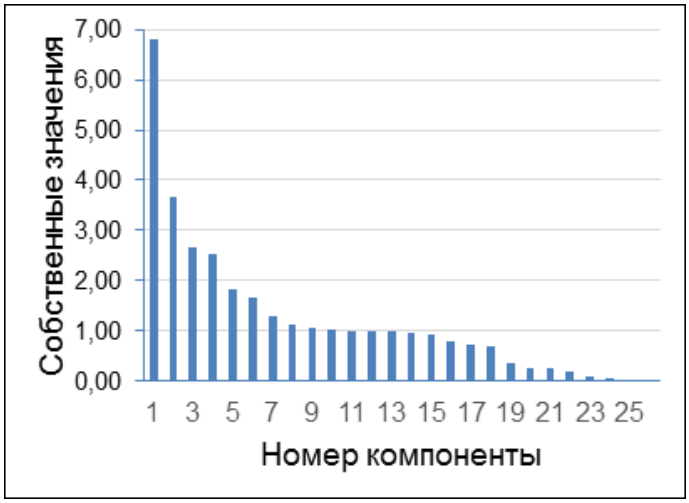

Рис. 2 – Распределение дисперсий по главным компонентам

Оценка качества PCA-модели была проведена на основе критерия адекватности выборки Кайзера-Мейера-Олкина. Величина, характеризующая степень применимости PCA к различным наборам статистических параметров производительности, находилась в диапазоне [0,71..0,84], что свидетельствует в пользу высокой (выше 0,8) и приемлемой (выше 0,7) пригодности метода к типичным статистическим выборкам. Так, расширение базового набора статистических параметров параметрами, специфичными для Oracle Exadata, такими как:

- io_interconnect_bytes – количество байт, пересланных между СУБД и системой хранения данных Exadata;
- io_offload_elig_bytes – количество байт, отфильтрованных для обработки на уровне Exadata;
- io_offload_return_bytes – количество байт, возвращенных с уровня Exadata;
- cell_uncompressed_bytes – количество байт, полученных при декомпрессии данных на узлах Exadata,

вопреки ожиданиям, не увеличило, а даже снизило оценки применимости PCA-метода.

Выбор числа компонент, используемых в создаваемой информационной модели SQL-запроса, определялся на основе различных критериев. Число используемых компонент согласно критерию Кайзера на различных наборах данных колебалось в пределах 10-12; при этом суммарный объем накопленной дисперсии приближается к 80 %. Для результатов, представленных в таблице 2, оно равно 11, поскольку именно первые 11 компонент выделяют дисперсии, первосходящие 1, а их суммарный объем равен 77,2%. Критерий «Сломанная трость» выделял 4-5 главных компонент, оставляя для дальнейшего анализа около 50% (49,04% или 54,78%) объяснённой дисперсии. Согласно критерию Кеттелла («Каменистая осыпь») информационная модель SQL-запроса включала от 6 до 9 первых компонент. Графическая интерпретация этого критерия для данных таблицы 2 представлена на рисунке 3. Убывание собственных значений после первых 8 компонент замедляется наиболее сильно, а сами отобранные компоненты содержат достаточно высокий процент объяснённой дисперсии (67,51%). С учетом того, что рекомендуемая к практическому применению при решении технических задач величина суммарной потери информации составляет не более 10-20%, а процедура отбора компонент при использовании стандартизации исходных данных достаточно просто формализована, был сделан выбор в пользу критерия Кайзера.

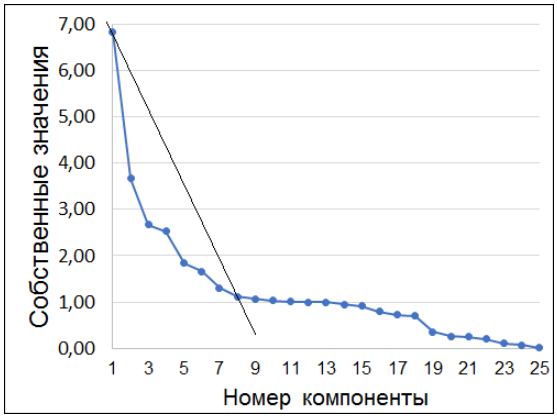

Рис. 3 – Визуальный отбор компонент по критерию Кеттелла

Предложенная модель SQL-запроса, сформированная на основе экспертных оценок статистических параметров производительности, методов корреляционного и факторного анализа, использована на этапе нейросетевой кластеризации для поиска ресурсоёмких и неэффективных запросов [1].

В качестве инструментов для проведённого анализа использовались авторские программные модули на языке программирования среды MatLab и языке Python с использованием библиотеки научных вычислений NumPy.

Список цитируемой литературы

1. Алгазали С.М.М., Айвазов В.Г., Кузнецова А.В. Совершенствование процесса поиска неэффективных SQL-запросов в СУБД Oracle // Интернет-журнал «Инженерный вестник Дона», №4, 2017, [Электронный ресурс]. ivdon.ru/ru/magazine/archive/n4y2017/4511 (дата обращения: 04.02.19).
2. Алгазали С.М.М., Кузнецова А.В. Кластеризация неэффективных SQL-запросов // Фундаментальные основы, теория, методы и средства измерений, контроля и диагностики: Матер. 19-ой Междунар. молод. науч.-практ. конф., г. Новочеркасск, ЮРГПУ(НПИ) – Новочеркасск: «Лик», 2018. – С.111 -117.