

УДК 004.65

<https://doi.org/10.33619/2414-2948/122/10>

ИССЛЕДОВАНИЕ И СРАВНЕНИЕ ПОДХОДОВ В АРХИТЕКТУРЕ ХРАНЕНИЯ ДАННЫХ, К ХЕШИРОВАНИЮ, ИНДЕКСАЦИИ И КОМПРЕССИИ ДАННЫХ В ORACLE И POSTGRES SQL

©Дубровская Е. А., ORCID: 0009-0002-9108-1506, Национальный исследовательский университет МЭИ, г. Москва, Россия, lena24042002@gmail.com

©Баланев К. С., ORCID: 0009-0002-9722-7262, SPIN-код: 8192-1861, Национальный исследовательский университет МЭИ, г. Москва, Россия, BalanevKS@mpei.ru

©Борцова Д. Э., ORCID: 0009-0006-3944-3089, SPIN-код: 7429-4020, Национальный исследовательский университет МЭИ, г. Москва, Россия, BortsovaDinE@mpei.ru

RESEARCH AND COMPARATIVE ANALYSIS OF APPROACHES TO DATA STORAGE ARCHITECTURE, HASHING, INDEXING, AND DATA COMPRESSION IN ORACLE AND POSTGRES SQL

©Dubrovskaya E., ORCID: 0009-0002-9108-1506, National Research University "Moscow Power Engineering Institute", Moscow, Russia, lena24042002@gmail.com

©Balanev K., ORCID: 0009-0002-9722-7262, SPIN-code: 8192-1861, National Research University "Moscow Power Engineering Institute", Moscow, Russia, BalanevKS@mpei.ru

©Bortsova D., ORCID: 0009-0006-3944-3089, SPIN-code: 7429-4020, National Research University "Moscow Power Engineering Institute", Moscow, Russia, BortsovaDinE@mpei.ru

Аннотация. Выполнен сравнительный анализ систем управления базами данных Oracle и PostgreSQL по основным архитектурным критериям: хранению данных, механизмам хеширования, индексации и компрессии. Цель исследования — выявить различия в архитектуре и оценить их влияние на производительность и масштабируемость. Показано, что Oracle реализует сложную многоуровневую модель хранения (tablespace–segment–extent–block), обеспечивающую эффективное управление пространством и высокую скорость доступа при любых типах нагрузок. PostgreSQL использует более простую страницу-ориентированную структуру (по 8 КБ на страницу), что упрощает администрирование, но снижает эффективность при аналитической обработке данных. При сравнении хеширования установлено, что Oracle благодаря механизму Hash Cluster обращается напрямую к нужному блоку данных (в среднем 0,5 мс на запрос), тогда как PostgreSQL требует дополнительного обращения к таблице после индексации, что увеличивает время доступа примерно вдвое. Обе СУБД используют B-tree-индексацию с логарифмической сложностью поиска, однако Oracle дополнительно применяет специализированные типы индексов (Bitmap, IOT), тогда как PostgreSQL делает акцент на расширяемость и поддержку пользовательских структур (GiST, GIN, BRIN). В части компрессии данных Oracle достигает сжатия до 8–10× (HCC, In-Memory Store) без потери производительности, тогда как PostgreSQL обеспечивает 3–5× за счёт TOAST и внешних расширений. Сравнительный анализ показал, что Oracle ориентирована на производительность и масштабируемость, а PostgreSQL — на гибкость и расширяемость архитектуры.

Abstract. The paper presents a comparative analysis of the Oracle and PostgreSQL database management systems based on key architectural criteria: data storage, hashing mechanisms, indexing, and data compression. The aim of the research is to identify architectural differences and assess their impact on performance and scalability. It is shown that Oracle implements a complex multi-level

storage model (tablespace–segment–extent–block), providing efficient space management and high data access speed under various workloads. PostgreSQL, in contrast, employs a simpler page-oriented structure (8 KB per page), which simplifies administration but reduces efficiency in analytical data processing. When comparing hashing mechanisms, it is established that Oracle, through the Hash Cluster feature, accesses the required data block directly (on average 0.5 ms per query), whereas PostgreSQL performs an additional table lookup after indexing, which approximately doubles access time. Both DBMSs utilize B-tree indexing with logarithmic search complexity; however, Oracle additionally employs specialized index types (Bitmap, IOT), while PostgreSQL focuses on extensibility and support for user-defined structures (GiST, GIN, BRIN). In terms of data compression, Oracle achieves compression ratios of up to 8–10× (using HCC and In-Memory Column Store) without performance loss, whereas PostgreSQL provides 3–5× compression through TOAST and external columnar extensions. The comparative analysis demonstrates that Oracle is oriented toward performance and scalability, while PostgreSQL emphasizes flexibility and architectural extensibility.

Ключевые слова: СУБД, Oracle, PostgreSQL, механизмы хеширования, компрессия данных.

Keywords: DBMS, Oracle, PostgreSQL, hashing mechanisms, and data compression.

Актуальность сравнения обусловлена тем, что Oracle и PostgreSQL реализуют классическую реляционную модель, поддерживают широкий спектр индексов, методов оптимизации и механизмов обеспечения целостности данных. Однако эти СУБД существенно различаются по внутренней архитектуре, структуре хранения и организации вычислений, что напрямую влияет на производительность и применимость в различных типах нагрузок. Также PostgreSQL активно развивается, осваивая технологии, ранее характерные для крупных корпоративных систем, включая колоночные расширения, усовершенствованные индексы и механизмы параллельной обработки, что делает сопоставление зрелой платформы Oracle и динамично адаптирующейся PostgreSQL особенно актуальным в современных условиях.

Критерии для сравнения СУБД Oracle и PostgreSQL. Oracle представляет собой коммерческую промышленную систему управления базами данных, ориентированную на крупные корпоративные среды и высоконагруженные системы, отличающуюся богатым набором функций и развитой инфраструктурой для обеспечения масштабируемости, отказоустойчивости и безопасности данных [1]. В отличие от неё, PostgreSQL — это открытая и активно развивающаяся реляционная СУБД, которая всё чаще используется как альтернатива коммерческим решениям благодаря высокой степени стандартизации SQL, расширяемости и поддержке как транзакционных (OLTP), так и аналитических (OLAP) сценариев обработки данных [3].

В работе рассматриваются такие критерии сравнения, как архитектура хранения данных, механизмы хеширования и компрессии данных. Архитектура хранения данных в Oracle основана на многоуровневой иерархии логических и физических структур, обеспечивающих эффективное управление пространством и оптимизацию ввода-вывода. Основными логическими структурами являются табличные пространства (tablespaces), сегменты (segments), экстенды (extents) и блоки данных (data blocks), их структуру можно увидеть на Рисунке 1.

Табличное пространство (Tablespace) — логическая область хранения, которая объединяет один или несколько физических файлов данных (datafiles). Каждый объект базы

данных хранится в определённом табличном пространстве. Формула 1 – это формальное представление логической структуры хранения в Oracle, выражающее связь между табличным пространством (tablespace) и физическими файлами данных (datafiles):

$$Tablespace = \sum_{i=1}^n Datafile_i \quad (1)$$

где *Tablespace* – Табличное пространство; *Datafile_i* – Файл данных с номером *i*, *n* – Общее количество файлов данных. Сегмент (Segment) — набор экстентов, выделенных для конкретного объекта базы данных. Например, таблица и индекс имеют собственные сегменты. Экстент (Extent) — непрерывная область блоков в файле данных, выделенная под хранение данных одного сегмента. Oracle динамически добавляет экстенты при росте данных. Блок данных (Data Block) — минимальная единица ввода-вывода. Размер блока может быть 2КБ, 4КБ, 8КБ и более, в зависимости от конфигурации системы. Все операции чтения и записи выполняются на уровне блока.

Oracle использует PCTFREE и PCTUSED параметры для контроля заполнения блоков, а также поддерживает Automatic Segment Space Management (ASSM) для оптимизации размещения строк. Дополнительно применяются механизмы row chaining и row migration для обработки строк, не помещающихся в один блок. При работе с большими таблицами и аналитическими запросами Oracle применяет Partitioning (разделение таблиц на логические части) и Hybrid Columnar Compression (HCC), где данные внутри блока хранятся в колонно-ориентированном виде, что улучшает сжатие и скорость выборки в OLAP-нагрузках [4].

PostgreSQL также использует многоуровневую структуру хранения, но её реализация проще и ориентирована на строко-ориентированную модель хранения [3].

Основные элементы: таблицы (tables), страницы (pages) и файлы сегментов. Таблица хранится в виде одного или нескольких файлов на диске. Каждый файл состоит из страниц фиксированного размера (по умолчанию 8 КБ). Формула 2 описывает структуру хранения данных таблицы в PostgreSQL. Она показывает, что таблица (Table) физически состоит из набора страниц (Page) — блоков фиксированного размера (по умолчанию 8 КБ), расположенных в одном или нескольких файлах на диске:

$$Table = \sum_{i=1}^n Page_i \quad (2)$$

где *Table* – таблица; *Page_i* – страница с номером *i*; *n* – количество страниц. Основными логическими структурами для PostgreSQL являются: “Table” — основной объект хранения данных, “File” — физический файл, содержащий страницы фиксированного размера (обычно 8 КБ), “Page” — базовая единица хранения, включающая заголовок, указатели строк и свободное пространство, “Tuple” — запись (строка) данных, содержащая атрибуты таблицы.

На Рисунке показана иерархическая модель хранения данных для двух СУБД — Oracle Database и PostgreSQL. Oracle имеет более сложную иерархическую модель хранения с чётким разграничением логических и физических структур, что обеспечивает гибкость и контроль над пространством, а также встроенные механизмы компрессии и оптимизации под разные нагрузки. PostgreSQL, напротив, использует более простую и компактную модель хранения на уровне страниц, что облегчает администрирование и повышает скорость операций при типичных OLTP-нагрузках, но требует внешних модулей для достижения уровня аналитических возможностей Oracle.

Механизмы хеширования в Oracle и PostgreSQL. Хеширование в обеих СУБД используется для ускорения поиска и выполнения соединений по равенству ключей, однако реализуется на разных уровнях архитектуры.

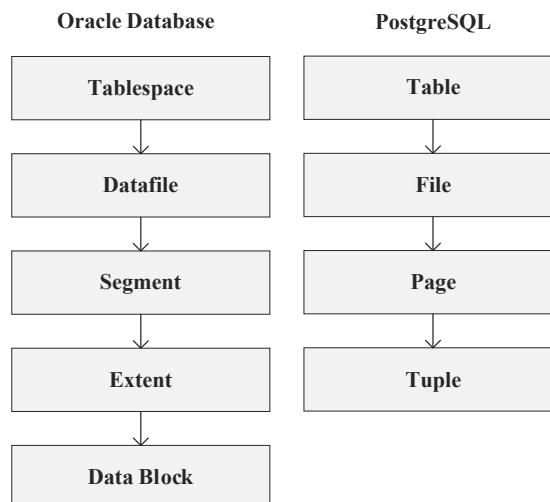


Рисунок. Иерархическая модель хранения данных для двух СУБД — Oracle Database и PostgreSQL

Для того чтобы количественно оценить различия в производительности, введём зависимость между временем поиска и количеством операций ввода-вывода:

$$t = IO \times \ell \quad (3)$$

где t — время поиска (в миллисекундах); IO — сколько раз нужно прочитать блок данных с диска; ℓ — время одного чтения блока (в среднем 0,5 мс для SSD). Эта формула позволяет связать архитектуру хранения данных с практическими показателями скорости выполнения запросов. В Oracle хеширование встроено прямо в физическую организацию данных через механизм Hash Cluster. Каждая строка таблицы помещается в особый блок — бакет, который выбирается по результату хеш-функции. Это значит, что при поиске строки по ключу Oracle сразу вычисляет значение хеша и обращается непосредственно к нужному блоку, без использования отдельного индекса. Если задать количество бакетов B примерно равным количеству страниц таблицы по формуле 4:

$$B \approx N / r_{pp} \quad (4)$$

где N - общее количество строк, r_{pp} — среднее количество строк на одной странице (около 80), то каждая группа строк (бакет) будет занимать один блок. Средняя нагрузка бакета считается по формуле 5:

$$\lambda = N / B \approx r_{pp} \quad (5)$$

где λ - среднее количество строк, приходящихся на один бакет (фактор загрузки). Если $\lambda \approx r_{pp}$, то каждая группа строк (бакет) будет занимать один блок, что обеспечивает оптимальное размещение данных и минимальное число обращений к диску. Таким образом, выполняется условие $IO_{Oracle} = 1$. Таким образом, время поиска составляет 0.5 миллисекунд и рассчитывается по формуле 6:

$$t_{Oracle} = 1 \times 0,5 = 0,5 \text{ мс} \quad (6)$$

В PostgreSQL данные не хранятся по хешам, а хеширование применяется только в индексах. Hash Index хранит вычисленные 32-битные хеши от значений ключей и ссылки на строки таблицы. При поиске PostgreSQL сначала находит нужный хеш в индексе (одно чтение), а затем выполняет *heap fetch* — обращение к таблице, чтобы достать саму строку (второе чтение). Если страница индекса содержит $pp \approx 200$ ссылок, то одно обращение к индексу и одно к таблице дают:

$$IO_{PostgreSQL} \approx 2 \Rightarrow t_{PostgreSQL} = 2 \cdot \ell = 2 \times 0,5 = 1,0 \text{ мс} \quad (7)$$

Пусть таблица содержит $N = 8000000$ строк, при среднем количестве строк на странице $r_{pp} = 80$. Тогда:

$$B \approx \frac{N}{r_{pp}} = \frac{8000000}{80} = 10000, \lambda = \frac{N}{B} = 80 \quad (8)$$

Это соответствует идеальному случаю: один блок на бакет, что обеспечивает только одно обращение к блоку данных (одна операция чтения с диска). Следовательно, при одинаковом времени чтения блока $\ell = 0,5$ мс:

$$t_{Oracle} = 0,5 \text{ мс}, t_{PostgreSQL} = 1,0 \text{ мс} \quad (9)$$

Проведённый анализ показывает, что Oracle выполняет поиск по хеш-ключу примерно в два раза быстрее, чем PostgreSQL, так как данные в Oracle хранятся непосредственно в блоках, определяемых хеш-функцией, в то время как PostgreSQL требует дополнительного обращения к таблице после поиска в индексе. Таким образом, интеграция механизма хеширования на уровне физического хранения обеспечивает Oracle преимущество по количеству операций ввода-вывода и общей производительности при точечных запросах.

Механизмы индексации в Oracle и PostgreSQL. Индексация является ключевым механизмом ускорения доступа к данным в реляционных СУБД, однако подходы к её реализации в Oracle и PostgreSQL различаются по архитектуре, степени интеграции с ядром и эффективности при разных типах нагрузок. Основным типом индекса в Oracle является B-tree (Balanced Tree) — сбалансированное дерево поиска, в котором каждый узел хранит отсортированные ключи и ссылки (ROWID) на строки таблицы. Такая структура обеспечивает логарифмическую сложность операций поиска, вставки и удаления: Oracle реализует многослойную и высокооптимизированную систему индексов, предназначенную для эффективной работы как в транзакционных (OLTP), так и в аналитических (OLAP) сценариях.

Индексы в Oracle представляют собой отдельные сегменты в физическом хранилище, управляемые независимо от таблиц, что обеспечивает гибкость при их настройке, перестроении и размещении в различных табличных пространствах. Основным типом индекса в Oracle является B-tree (Balanced Tree) — сбалансированное дерево поиска, в котором каждый узел хранит отсортированные ключи и ссылки (ROWID) на строки таблицы. Такая структура обеспечивает логарифмическую сложность операций поиска, вставки и удаления:

$$t = O(\log_2 n) \quad (10)$$

где n — количество записей в индексе. B-tree индекс позволяет находить данные значительно быстрее, чем при последовательном просмотре таблицы. Поскольку ключи в дереве всегда упорядочены, B-tree индекс одинаково эффективен как для точечных запросов ($=$), так и для диапазонных выборок ($>$, $<$, BETWEEN, LIKE). Oracle B-tree индекс обеспечивает равномерное распределение ключей и минимальную глубину дерева (в среднем

3–5 уровней), что гарантирует стабильное время доступа даже при обработке миллионов записей. Кроме того, Oracle поддерживает специализированные типы индексов — Bitmap, Function-based, Reverse Key и Index-Organized Tables (IOT), выбор которых зависит от характера нагрузки и типа обрабатываемых данных.

PostgreSQL реализует открытую и расширяемую архитектуру индексов, ориентированную на гибкость и адаптацию под разные типы данных и сценарии использования. Каждый индекс в PostgreSQL является отдельным физическим объектом, синхронизированным с таблицей через механизмы MVCC (Multiversion Concurrency Control) и Visibility Map, что обеспечивает корректность выборки при параллельных транзакциях без блокировок чтения. Основным типом индекса в PostgreSQL также является B-tree, реализующий классическую модель сбалансированного дерева поиска, в котором каждый узел содержит ключ и ссылку на физический идентификатор строки (TID).

Как и в Oracle, операции поиска, вставки и удаления имеют логарифмическую сложность, рассчитанную по формуле 10. однако структура индекса тесно связана с механизмом хранения таблицы (heap), из-за чего при обращении к данным всегда выполняется дополнительное чтение страницы таблицы после нахождения нужного ключа.

Особенностью PostgreSQL является расширяемость системы индексов — разработчики могут создавать собственные типы индексов через модульную инфраструктуру доступа к методам (Access Method API). Помимо стандартного B-tree, PostgreSQL поддерживает и активно использует специализированные типы индексов: Hash Index — ускоряет поиск по равенству (=); GiST (Generalized Search Tree) — для геометрических, пространственных и полнотекстовых данных; GIN (Generalized Inverted Index) — для JSON, массивов и документов; BRIN (Block Range Index) — для больших таблиц с последовательными диапазонами значений. Oracle обеспечивает высшую производительность и минимальные I/O-затраты за счёт глубокой интеграции индексов в архитектуру хранения, тогда как PostgreSQL делает ставку на гибкость и расширяемость, позволяя пользователю выбирать или создавать тип индекса под конкретную задачу, пусть даже с небольшой потерей эффективности при прямом доступе к данным.

Механизмы компрессии данных в Oracle и PostgreSQL. Oracle реализует многоуровневую систему компрессии данных, встроенную в архитектуру хранения, что обеспечивает высокую степень сжатия без заметной потери производительности. Основные технологии:

Table Compression (Basic / OLTP) — сжимает повторяющиеся значения на уровне блоков, снижая размер хранилища до 2–3 раз.

Hybrid Columnar Compression (HCC) — комбинирует построчное и колоночное хранение, обеспечивая сжатие до 10:1 для аналитических таблиц за счёт группировки данных по колоннам. In-Memory Column Store использует Run-Length Encoding (RLE) и Dictionary Encoding для колоночных сегментов, что позволяет хранить большие объёмы данных в оперативной памяти без разжатия при запросах. Формально эффективность рассчитывается по формуле 11.

$$C = \frac{S_{orig}}{S_{comp}} \quad (11)$$

где C — коэффициент сжатия, S_{orig} — исходный объём, S_{comp} — объём после компрессии. Для технологии гибридного колоночного сжатия данных HCC (Hybrid Columnar Compression) $C_{Oracle} \approx 8 - 10$ [6]. Главное преимущество такого механизма — компрессия интегрирована на уровне блока и столбца, поддерживается при чтении без распаковки, снижая I/O-затраты.

PostgreSQL использует более простую, построчную модель компрессии, реализованную на уровне хранения строк и внешних значений. Основные механизмы: TOAST (The Oversized-Attribute Storage Technique) — автоматически сжимает большие поля (TEXT, BYTEA, JSONB) с помощью алгоритма pglz (вариант LZ-компрессии); Columnar extensions — внешние модули, такие как cstore_fdw, Zedstore и TimescaleDB, реализуют колоночное хранение и поддержку RLE, Dictionary Encoding и Delta Encoding, обеспечивая сжатие до 4–6 раз для аналитических нагрузок.

Для PostgreSQL коэффициент сжатия оценивается как $C_{PostgreSQL} \approx 3 - 5$ в зависимости от используемого метода и однородности данных. Особенность — компрессия выполняется при записи, но требует разжатия при чтении, что может увеличить нагрузку на CPU при сложных запросах [7].

Oracle обеспечивает значительно более эффективную и «прозрачную» компрессию, встроенную в ядро СУБД и не влияющую на время выборки, тогда как PostgreSQL опирается на внешние механизмы и построчное сжатие, что делает его более гибким, но менее эффективным при больших аналитических наборах данных. Проведённый анализ показал, что Oracle и PostgreSQL, реализуя общие принципы реляционной модели, существенно различаются по архитектурным подходам, влияющим на производительность, масштабируемость и применимость в различных типах нагрузок. Oracle демонстрирует высокую степень интеграции механизмов хранения, индексации, хеширования и компрессии на уровне ядра, что обеспечивает минимальные затраты ввода-вывода, устойчивую работу с большими аналитическими наборами данных и высокий коэффициент сжатия без потери скорости обработки. PostgreSQL, напротив, отличается открытой и модульной архитектурой, позволяющей адаптировать систему под конкретные задачи за счёт расширяемости индексов, поддержки внешних модулей компрессии и гибкой настройки параметров хранения. При этом упрощённая модель страниц и независимость от проприетарных технологий делают PostgreSQL более удобным решением для разработчиков и малых систем с OLTP-нагрузкой. Таким образом, Oracle оптимальна для корпоративных и аналитических систем, требующих высокой производительности и устойчивости, тогда как PostgreSQL предпочтительна в средах, где важны гибкость, открытость и экономическая эффективность, что определяет их нишевую специализацию и взаимодополняемость в современной практике построения баз данных.

Источники:

1. Oracle Database Concepts. Oracle Documentation. clck.ru/3R9PV2
2. PostgreSQL Official Documentation. <https://clck.ru/3R9PVg>
3. Oracle Hybrid Columnar Compression Overview. <https://clck.ru/3R9PW4>
4. PostgreSQL Documentation: Chapter 63. Database Physical Storage. <https://clck.ru/3R9PWk>
5. PostgreSQL TOAST Mechanism. <https://clck.ru/3R9PX5>
6. Oracle Corporation. Hybrid Columnar Compression Technical White Paper, 2020
7. PostgreSQL Documentation 16 – TOAST Storage and Compression. <https://clck.ru/3R9PX5>

Sources:

1. Oracle Database Concepts. Oracle Documentation. clck.ru/3R9PV2
2. PostgreSQL Official Documentation. <https://clck.ru/3R9PVg>
3. Oracle Hybrid Columnar Compression Overview. <https://clck.ru/3R9PW4>

4. PostgreSQL Documentation: Chapter 63. Database Physical Storage.
<https://clck.ru/3R9PWk>
5. PostgreSQL TOAST Mechanism. <https://clck.ru/3R9PX5>
6. Oracle Corporation. Hybrid Columnar Compression Technical White Paper, 2020
7. PostgreSQL Documentation 16 – TOAST Storage and Compression. <https://clck.ru/3R9PX5>

Поступила в редакцию
12.11.2025 г.

Принята к публикации
19.11.2025 г.

Ссылка для цитирования:

Дубровская Е. А., Баланев К. С., Борцова Д. Э. Исследование и сравнение подходов в архитектуре хранения данных, к хешированию, индексации и компрессии данных в Oracle и PostgreSQL // Бюллетень науки и практики. 2026. Т. 12. №1. С. 76-83.
<https://doi.org/10.33619/2414-2948/122/10>

Cite as (APA):

Dubrovskaya, E., Balanov, K., & Bortsova, D. (2026). Research and Comparative Analysis of Approaches to Data Storage Architecture, Hashing, Indexing, and data Compression in Oracle and PostgreSQL. *Bulletin of Science and Practice*, 12(1), 76-83. (in Russian).
<https://doi.org/10.33619/2414-2948/122/10>