# Оптимизация БД

Тлеубаева А.О.

# Введение

- Оптимизация производительности баз данных (БД) занимает ключевое место в управлении базами данных, но часто остается недооцененной в учебных курсах. В университетах обычно используются БД с небольшим количеством записей, что снижает акцент на эффективность запросов, поскольку даже неэффективные запросы показывают приемлемую производительность при малом объеме данных. Однако в реальных условиях, когда объемы данных исчисляются миллионами записей, низкая эффективность может привести к неприемлемо долгому времени выполнения запросов.

# Основная цель БД

Основная функция системы управления базами данных (СУБД) - предоставление своевременных ответов пользователям, которые взаимодействуют с системой следующим образом:

1.Клиентское приложение формирует SQL-запрос.

2.Запрос отправляется на сервер (СУБД).

3.Сервер обрабатывает запрос и формирует результат.

4.Результаты отправляются обратно клиенту.

Пользователи ожидают быстрого возврата результатов, однако оценить "хорошую" производительность сложно, в отличие от "плохой", которая обычно выявляется через жалобы на медленные запросы. Поэтому СУБД требуется регулярная оптимизация для улучшения времени ответа.

# Ключевые аспекты производительности

Производительность БД ограничивается тремя основными ресурсами:

- **Вычислительная мощность CPU**,
- **Оперативная память**,
- **Пропускная способность ввода/вывода** (диски и сеть).

Эффективное использование этих ресурсов требует их оптимизации, чтобы избежать узких мест в производительности.

# Оптимизация клиент-сервер

Оптимизация делится на действия по стороне клиента и сервера:

- **На стороне клиента**: цель - сформировать запрос, который быстро и эффективно обрабатывается сервером.
- **На стороне сервера**: СУБД должна быть настроена так, чтобы максимально быстро обрабатывать запросы, оптимально используя доступные ресурсы.

# Архитектура СУБД

Архитектурно СУБД включает в себя:

- **Файлы данных**: основное хранилище данных.
- **Кэш данных** и **SQL кэш**: для ускорения доступа к данными и SQL операциям.
- **Процессы СУБД**: включая обработчик запросов, планировщик, менеджер блокировок и оптимизатор запросов.

## Режимы оптимизации запросов

Оптимизация может быть **автоматической** или **ручной**, и может происходить на этапе компиляции (статическая) или во время выполнения (динамическая). Выбор метода оптимизации зависит от текущих потребностей и архитектуры приложения.

## Статистика БД

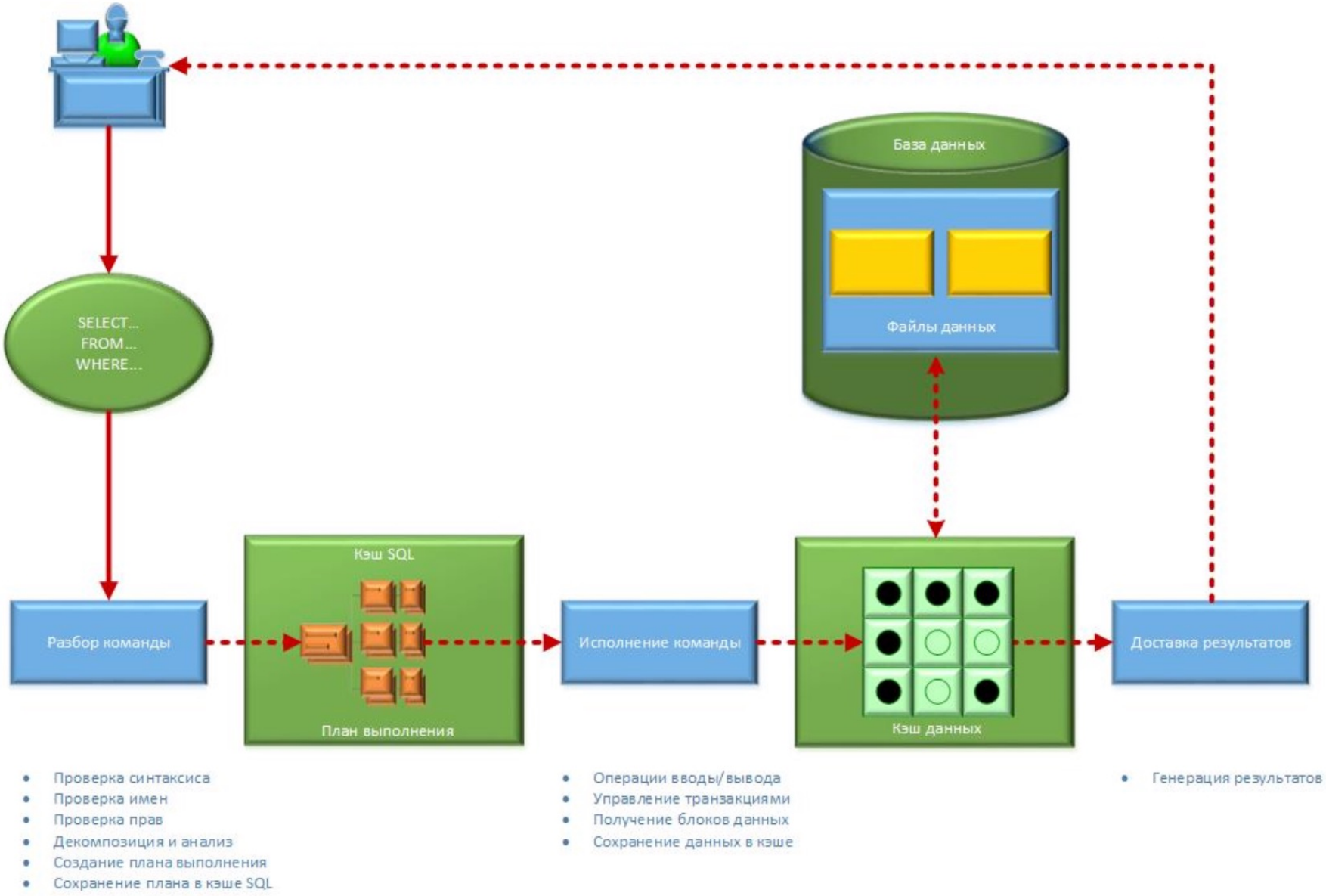
- Статистика по объектам БД играет критическую роль в оптимизации, поскольку предоставляет данные о размере таблиц, количестве записей, блоках данных и индексах. Обновление статистики может быть автоматизировано или выполняться вручную, в зависимости от СУБД (например, команды ANALYZE в Oracle или UPDATE STATISTICS в SQL Server).

Оптимизация баз данных — это комплексная и многоуровневая задача, требующая постоянного внимания к деталям, как в плане аппаратного, так и программного обеспечения. Правильный подход к настройке и обновлению статистики, а также к проектированию архитектуры БД может значительно улучшить производительность и сократить время отклика на запросы пользователей.

# Обработка запросов

Обработка SQL-запросов в системах управления базами данных (СУБД) происходит в три основных этапа:

- **1. Разбор**
- В этой фазе СУБД анализирует SQL-запрос и формирует наиболее эффективный план его выполнения. Этапы разбора включают:
- **Проверку синтаксиса:** Убедиться, что запрос корректно сформулирован.
- **Валидация объектов данных:** Проверка наличия таблиц и столбцов, упомянутых в запросе.
- **Проверка прав доступа:** Убеждение в том, что у пользователя есть необходимые права для выполнения запроса.
- **Декомпозиция запроса:** Разложение запроса на атомарные компоненты.
- **Оптимизация запроса:** Преобразование запроса в более эффективную форму, сохраняя его семантическую эквивалентность.
- На основании доступной статистики по базе данных оптимизатор выбирает лучший план доступа, который затем кэшируется для повторного использования.




# Выполнение запроса

# Исполнение

На этом этапе СУБД реализует план выполнения запроса. Процесс включает:

- **Выполнение ввода-вывода:** Операции чтения данных из физического хранилища.
- **Блокировки:** При необходимости устанавливаются блокировки для управления параллельным доступом.
- **Транзакционное управление:** Обеспечение целостности данных в процессе выполнения запроса.

# Доставка

Последний этап обработки запроса заключается в формировании и отправке результата клиенту. СУБД может использовать временные хранилища для обработки данных, осуществляя сортировку, группировку или агрегацию. Результаты последовательно отправляются клиенту, начиная с первых строк.

# Узкие места обработки запросов

Основные узкие места включают:

- **Центральный процессор (ЦП):** Недостаточная мощность ЦП может замедлять обработку, особенно при высокой нагрузке.
- **Оперативная память (ОЗУ):** Недостаток ОЗУ приводит к частым операциям подкачки, замедляя систему.
- **Жесткий диск:** Медленные операции ввода-вывода увеличивают время доступа к данным.
- **Сеть:** Ограниченная пропускная способность сети может стать причиной задержек при обмене данными между клиентом и сервером.
- **Код приложения:** Плохо написанный код или неоптимальные запросы могут усугублять проблемы производительности.
- Эффективная оптимизация запросов требует учета всех этих аспектов, чтобы минимизировать задержки и ускорить выполнение операций.

На следующем этапе мы перейдем к детальному рассмотрению индексов и их роли в оптимизации запросов, обеспечивая быстрый доступ к данным и минимизируя необходимость полных сканирований таблиц.

# Выбор оптимизатора

Оптимизация запросов является ключевым этапом в процессе обработки SQL-запросов в СУБД. Оптимизатор запросов — это компонент СУБД, который определяет наиболее эффективный способ выполнения запроса. Оптимизаторы бывают двух типов:

**Оптимизатор на основе правил**

- Использует заранее заданные правила для оценки планов выполнения запросов.
- Каждое действие или операция имеет фиксированную "стоимость".
- Сумма стоимостей действий определяет общую стоимость плана выполнения.

**Оптимизатор на основе затрат**

- Применяет более сложные алгоритмы, которые учитывают статистические данные о БД.
- Рассчитывает стоимость обработки, затраты на ввод-вывод и использование ресурсов (ОЗУ, временное пространство).
- Цель — минимизировать общую стоимость выполнения запроса.

# Настройка производительности SQL

- Настройка SQL осуществляется с учётом того, как написан запрос:

**Индексы и их селективность**

- Индексы ускоряют доступ к данным, особенно если они присутствуют в условиях поиска (WHERE, HAVING), сортировке (ORDER BY) или при использовании агрегатных функций (MAX, MIN).
- Селективность индекса — это вероятность его использования при выполнении запроса.
- Следует создавать индексы для столбцов, часто используемых в условиях поиска, но избегать их в маленьких таблицах или таблицах с низкой разреженностью данных.

**Общие рекомендации по индексации**

- Индексировать столбцы, используемые в WHERE, HAVING, ORDER BY, и GROUP BY.
- Не индексировать каждый столбец в таблице, чтобы избежать негативного влияния на операции вставки, обновления и удаления.
- Объявлять первичные и внешние ключи для использования в соединениях.

# Формулировка запроса

При формулировке запроса следует учитывать:

- **Необходимые столбцы и вычисления**: определить, какие данные нужны и какие вычисления должны быть выполнены.
- **Исходные таблицы**: выбрать таблицы, которые содержат требуемые данные.
- **Способ соединения таблиц**: определить, как лучше всего организовать соединение между таблицами.
- **Критерии выбора**: задать условия выбора данных.
- **Порядок отображения данных**: использовать ORDER BY для упорядочивания результатов.

# Настройка производительности СУБД

Настройка СУБД включает:

- **Управление кэшем данных и SQL**: оптимизация размера кэшей для улучшения доступа к часто используемым данным.
- **Режим оптимизатора**: выбор между оптимизацией на основе затрат и на основе правил в зависимости от доступности статистики.
- **Физическое хранение данных**: использование RAID, выбор конфигураций для оптимизации производительности и отказоустойчивости.

# Применение технологий хранения

- **Использование SSD и RAID** для ускорения операций ввода-вывода и повышения надёжности.
- **Организация хранения данных**: оптимальное распределение данных и индексов по различным табличным пространствам и физическим носителям для минимизации конфликтов и ускорения доступа.
- Эти методы настройки и оптимизации помогают значительно повысить эффективность работы базы данных, уменьшая время отклика на запросы и повышая общую производительность системы.

# Итоги

## Основные Понятия

## 1.Настройка Производительности Базы Данных:

1. **Настройка производительности SQL:** Включает в себя действия на стороне клиента для генерации эффективного кода SQL, минимизирующего время ответа и ресурсы сервера.
2. **Настройка производительности СУБД:** Охватывает настройки на стороне сервера, направленные на оптимизацию реакции СУБД на клиентские запросы при использовании существующих ресурсов.

## 2.Статистика Базы Данных:

1. Описывает текущее состояние объектов базы данных (таблиц, индексов) и ресурсов (процессоры, память), используемая для принятия решений об оптимизации обработки запросов.

## 3.Этапы Обработки Запросов:

1. **Синтаксический Анализ:** Выбор наиболее эффективного плана доступа.
2. **Исполнение:** Реализация выбранного плана.
3. **Выборка:** Сбор и отправка данных клиенту.

# Итоги

**5. Индексы:**

1. Ключевой элемент для ускорения доступа к данным, особенно полезен в столбцах с высокой разреженностью данных.

**6. Оптимизация Запросов:**

1. Включает выбор индексов, методов соединения таблиц, и определение последовательности обработки таблиц. Осуществляется через оптимизаторы на основе правил или затрат.

**7. Формулировка Запроса:**

1. Преобразование бизнес-запросов в SQL-код, требующее тщательного анализа необходимых столбцов, таблиц и вычислений.

**8.Настройка Производительности СУБД:**

1. Управление памятью и процессами для оптимизации обработки запросов и хранения данных.

# Контрольные вопросы

1. Что такое оптимизатор запросов и какие существуют его основные типы?
2. Какие факторы влияют на выбор плана выполнения запроса оптимизатором?
3. Какие основные методы настройки производительности SQL вы знаете?
4. Что такое индекс и каковы принципы его эффективного использования?
5. Какие общие ошибки могут снижать производительность SQL-запросов?
6. В чем заключается важность физической организации данных в СУБД?
7. Как можно использовать RAID в контексте баз данных для повышения производительности и надежности?