Hive

Tags

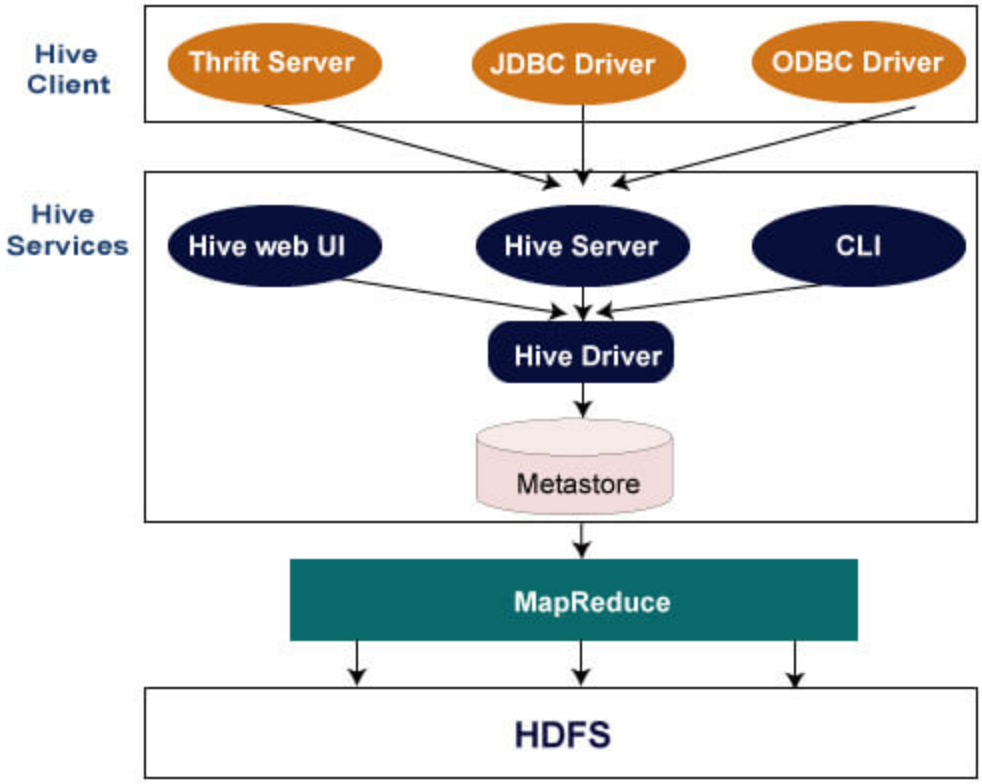
Hive – это **инструмент для хранения и анализа данных**, работающий поверх Hadoop и использующий HDFS (Hadoop Distributed File System) для хранения больших объемов данных. **Apache Hive не является базой данных** в традиционном смысле.

Основная цель – **упрощение работы с большими объемами данных** через SQL-подобный язык запросов, называемый **HiveQL** (Hive Query Language).

Hive переводит запросы на **HiveQL** в **MapReduce**, **Apache Tez** или **Spark** задачи, которые затем выполняются в кластере Hadoop.

▼ Особенности и Архитектура Hive

- **Hive Metastore**: хранит информацию о структуре таблиц и данных, включая схемы таблиц, местоположение данных, разделы и т.д. (часто используют RDBMS для хранения метаданных). **Derby** — это стандартная база данных для **Apache Hive Metastore**, поставляемая вместе с Hive. Для **больших развертываний** рекомендуется использовать внешнюю базу данных для Metastore, например **MySQL, PostgreSQL, Oracle** или **Microsoft SQL Server**. Чтобы использовать внешнюю базу данных, необходимо настроить Hive для использования этой базы и создать в ней нужные таблицы с помощью инструмента Hive Schema Tool.
- **HiveQL**: SQL-подобный язык для выполнения запросов.
- **Execution Engine**: отвечает за выполнение плана запроса, трансформируя HiveQL в задачи для MapReduce, Tez или Spark.
- **Driver**: управляет сессией, компилирует и оптимизирует запросы, создает план выполнения.
- **CLI, Thrift Server и JDBC/ODBC**: методы доступа для пользователей.



▼ Преимущества и недостатки

Преимущества:

- Простота написания SQL-подобных запросов.
- Поддержка огромных объемов данных.
- Гибкость в использовании различных движков выполнения.
- Расширяемость через UDF/UDAF/UDTF.

Недостатки:

- Высокая задержка выполнения запросов (особенно с MapReduce). **Hive не поддерживает Catalyst Optimizer напрямую**, но использует свой собственный оптимизатор.
- Ограниченная поддержка ACID.
- Не поддерживает OLTP (только OLAP).

▼ Основные компоненты Hive

Таблицы (Tables):

Internal Tables (или **управляемые таблицы**) в Apache Hive — это таблицы, для которых Hive **сам управляет данными и метаданными**. Эти таблицы отличаются от внешних (External Tables) тем, что Hive сам контролирует их жизненный цикл, а именно: при удалении таблицы данные в HDFS также удаляются. Данные Internal Tables хранятся в `/user/hive/warehouse`

External Tables (внешние таблицы): Hive управляет только схемой, а данные хранятся за пределами Hive. При удалении таблицы данные остаются. При обновлении через Hive файл в HDFS не изменится, если вы сделаете `UPDATE` в Hive для внешней таблицы. Hive не поддерживает изменения файлов для внешних таблиц (external tables) в HDFS при использовании команд `UPDATE` или `DELETE`.

- **Partitioned Tables (разделенные таблицы)**: Данные в этих таблицах разбиваются на части по одному или нескольким столбцам. Разделение улучшает производительность запросов, поскольку уменьшает объем данных, которые необходимо сканировать.
- **Bucketed Tables (корзинированные таблицы)**: Bucketing (корзинирование) — это техника для повышения производительности запросов, которая делит большие таблицы на небольшие управляемые части. Как и при разделении, корзинирование позволяет распределить данные на более мелкие, равномерно распределенные группы.
- **Virtual Tables (виртуальные таблицы)**: Виртуальные таблицы создаются с помощью HiveQL и не хранят данные на диске. Вместо этого они представляют собой виртуальное представление одной или нескольких существующих таблиц.
- **Temporary Tables (временные таблицы)**: Временные таблицы создаются и управляются в рамках одной сессии Hive. Эти таблицы

автоматически удаляются по завершении сессии.

Базы данных (Databases):

- Логическая группировка таблиц, чтобы упростить организацию данных и разграничение доступов.

Разделы (Partitions):

- Логическое деление таблиц по ключевым колонкам для улучшения производительности. Например, можно делить данные по дате.

Кластеризация/разбиение (Buckets):

- Дополнительное деление разделов для улучшения производительности запросов. Используется для мелкозернистой организации данных.

▼ Bucketing

Bucketing — это техника в Hive для разделения данных в таблице на управляемые и более эффективные наборы файлов. Она применяется для оптимизации запросов и управления большими наборами данных, распределяя данные по фиксированному количеству корзин (buckets) на основе значения хеш-функции, примененной к определенной колонке в таблице.

Каждая корзина хранится как отдельный файл, что позволяет Hive более эффективно читать и записывать данные, используя ключ разбиения.

Как работает Bucketing?

- **Хеш-функция** применяется к значению определенного столбца (bucketing key), чтобы вычислить, в какую корзину попадет запись.
- Результат деления хеш-значения на общее количество корзин определяет, в какой файл (корзину) будет помещена каждая запись.

- В отличие от **разделения (partitioning)**, которое создает директории на основе значений конкретных столбцов, bucketing создает **фиксированное количество файлов** на основе хеш-значения столбца.

Преимущества и полезные аспекты Bucketing

1. **Снижение объема ввода-вывода (I/O)**:
    - Bucketing помогает сократить объем I/O операций за счет разделения данных на фиксированное количество файлов, что позволяет уменьшить количество файлов, которые необходимо сканировать при запросах. Это ускоряет выполнение запросов, ограничивая объем данных, подлежащих чтению.
2. **Борьба с дисбалансом данных (Data Skew)**:
    - Корзинирование позволяет **равномерно распределить данные** по корзинам, что особенно полезно для таблиц с высоким уровнем дисбаланса данных. Равномерное распределение снижает вероятность создания «горячих точек» (hotspots), которые могут замедлить выполнение запросов.
3. **Повышение производительности при объединении (Join performance)**:
    - При объединении таблиц по ключу корзинирования Hive может выполнять **map-side join** вместо традиционного **reduce-side join**. Это намного быстрее, так как Hive может читать данные напрямую из файлов корзин, без необходимости сортировать и передавать данные по сети, что значительно ускоряет процесс объединения.
4. **Эффективное создание выборок (Sampling)**:
    - Bucketing также можно использовать для более эффективного создания выборок. Выбирая только часть корзин, а не сканируя всю таблицу, Hive может быстро вернуть **репрезентативную выборку данных**, что полезно для аналитических задач и тестирования.

Bucketing в сочетании с другими функциями Hive

Bucketing можно комбинировать с другими возможностями Hive, такими как:

- **Partitioning**: вместе с корзинированием разбиение данных позволяет еще больше сократить объем данных для сканирования.
- **Индексация**: ускоряет поиск нужных данных в корзинированных таблицах.
- **Сжатие**: уменьшает объем хранимых данных, сохраняя при этом высокую производительность.

Bucketing особенно полезен для больших таблиц или таблиц с дисбалансом данных и может значительно повысить общую производительность и эффективность работы с данными в Hive.

▼ HQL vs SQL

- HiveQL отличается от стандартного SQL, так как направлен на работу с данными в Hadoop, но поддерживает многие SQL-конструкции.
- Hive не поддерживает транзакции и ограниченные подзапросы.
- Поддержка OLAP-операций, но не подходит для OLTP (онлайн-транзакционной обработки).

▼ Безопасность и управление доступом в Hive

- **Аутентификация**: Kerberos обычно используется для обеспечения аутентификации в кластерах Hadoop.
- **Авторизация**: разграничение доступа можно настроить через Apache Ranger, Sentry или стандартные HDFS ACLs.
- **Шифрование данных**: может быть настроено на уровне HDFS для защиты данных на диске.

▼ Маленькие темы

▼ Что такое файл .hiverc?

Файл **".hiverc"** — это файл конфигурации Hive, который задает настройки и параметры по умолчанию для оболочки Hive. При запуске

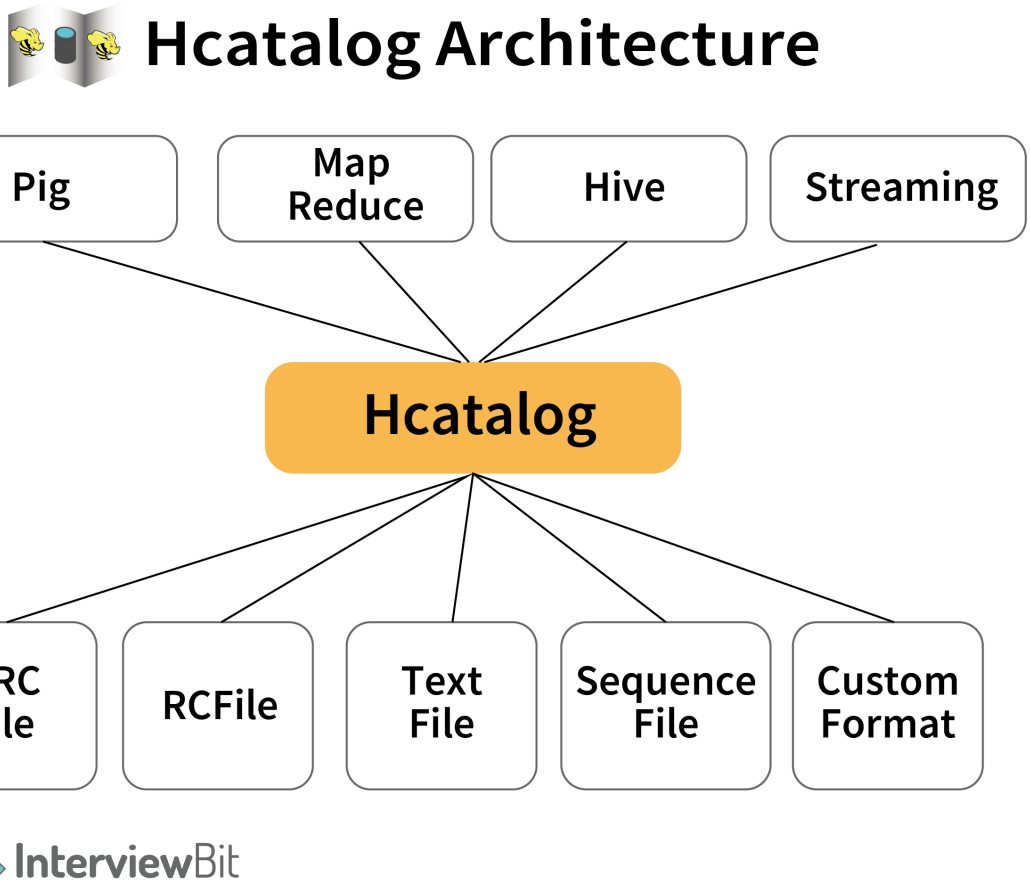
Hive считывает файл ".hiverc" и применяет указанные в нем параметры к текущей сессии оболочки Hive.

- **Создание алиасов**: для экономии времени и повышения продуктивности можно создавать алиасы для часто используемых команд Hive. Например, можно задать алиас для команды `SELECT * FROM table` как "sel *".
- **Установка переменных окружения**: файл ".hiverc" может быть использован для установки переменных окружения, которые используются Hive, таких как `HADOOP_HOME` и `HIVE_CON_DIR`.

▼ Что такое HCatalog?

HCatalog — это компонент проекта Apache Hive, который служит уровнем управления хранилищем и таблицами для Hadoop-кластеров.

HCatalog предоставляет инструменты для доступа к таблицам Hive Metastore из других фреймворков, таких как **Pig**, **Spark SQL** и пользовательские приложения на базе **MapReduce**.
HCatalog предоставляет
единый слой схем и метаданных для данных в Hadoop, что позволяет унифицировать доступ к данным и управлять ими.



▼ Как посмотреть индексы в HIVE

Чтобы просмотреть индексы, созданные для таблицы в Hive, можно использовать следующую команду:

```
SHOW INDEXES ON table_name;
```