**Дистрибуция и создание таблиц**

Раздаточный материал ко 2-му уроку, посвященному дистрибуции и созданию таблиц.

|  |  |
| --- | --- |
| СОДЕРЖАНИЕ | |
| 1 | Дистрибуция |
| 2 | Ориентация таблиц |
| 3 | Констрейнты |
| 4 | Партиционирование |

Дистрибуция

Дистрибуция или распределение данных. Это один из важнейших факторов быстродействия системы. В Greenplum обработка запроса на сегментах выполняется в параллель, скорость выполнения запроса равняется скорости выполнения запроса на самом медленном сегменте.

Каждая таблица в Greenplum хранится не на одном, а сразу на нескольких сегментах. Каждый сегмент хранит свою порцию данных. Если данные распределены неравномерно, то сегменты с большим количеством данных будут выполнять работы дольше, что приводит к падению производительности всей системы. Поэтому необходимо стремиться к равномерному распределению данных по всем сегментам системы.

Для каждой таблицы задается своя политика дистрибуции – способ, по которому строки распределяются по сегментам. Он задается при создании таблицы в команде CREATE TABLE. А также может быть изменен в последствие командой ALTER TABLE

Ключ дистрибуции – поле или набор полей по значению которого определяется на каком сегменте будет храниться строка таблицы

При выборе ключа дистрибуции необходимо учитывать 2 основных момента.

1. Ключ должен обеспечивать равномерность распределения данных
2. Обеспечение локальности операции. Под локальными операциями подразумеваются ситуации, когда соединяемые таблицы с помощь. Конструкции join таблицы имеют одинаковый ключ дистрибуции, что позволяет выполнять операции локально на сегментах без необходимости перераспределять данные

В Greenplum существует 3 вида дистрибуции:

* DISTRIBUTED BY
* DISTRIBUTED RANDOMLY
* DISTRIBUTED REPLICATED

DISTRIBUTED BY - Дистрибуция по HASH указанных полей. В этом случае из значения указанного поля или полей вычисляется хэш по которому в дальнейшем и определяется на какой сегмент попадет та или иная запись

DISTRIBUTED RANDOMLY – Планировщик автоматически распределяет данные по сегментам. Несмотря на название данные распределяются не случайным образом, а последовательно. Первая запись, записывается на первый сегмент. Вторая запись на второй и т. д. Поэтому важно следить, чтобы записи в таблицы с таким распределением, вставлялись большими пакетами. если записи будут вставляться в таблицу небольшими количествами, например по одной строке, то все они будут записаны на первый сегмент, что приведет к перекосу данных.

DISTRIBUTED REPLICATED – на каждом сегменте хранится полная копия таблицы. Такой метод подходит для небольших таблиц (справочников).

При выборе ключа необходимо придерживаться следующих правил

1. Поле должно обладать высокой селективностью (идентификаторы). Т. е. иметь большое количество уникальных значений
2. В ключе не должно быть null значений или значений по умолчанию. В противном случае, все null значения попадут на один сегмент, что приведет к перекосу данных
3. Перекос данных может быть вызван неравномерным распределением данных из-за неправильного выбора ключей распределения или операций вставки.
4. Присутствие на уровне таблиц перекоса данных часто является основной причиной низкой производительности запросов и нехватки памяти.
5. Нужно следить за перекосом данных иначе запрос будет работать со скоростью самого перегруженного сегмента.
6. Проверить распределение по сегментам можно, используя служебный столбец gp\_segment\_id, который присутствует в каждой таблице.

Изменить ключ распределения можно двумя способами.

1. Первый способ — это изменение без пересоздания таблицы с помощью конструкции ALTER TABLE SET.
2. Второй способ позволяет перераспределить данные по прежнему ключу дистрибуции при помощи опции REORGANIZE TRUE.

Используется в двух случаях:

* 1. Добавление нового сегмента
  2. Для избежания раздутия таблиц

Общие рекомендации при создании таблиц.

* Явно задавайте способ и ключ распределения при создании таблицы, в том числе для временных таблиц. Если этого не делать, то ключом распределения будет назначен первый атрибут таблицы.
* Используйте RANDOMLY распределение для небольших таблиц где нет хороших кандидатов на ключ распределения из одного атрибута. Выбирать 2 и более поля в качестве ключа дистрибуции не рекомендуется, т.к. в таком случае вычисление хэша займет длительное время

Ориентация таблиц

При создании таблиц необходимо полнить то, как именно мы будем работать с данными, которые хранятся в таблице от этого зависит оптимальный метод их хранения.

Greenplum поддерживает 2 метода хранения heap и Append Optimize.

* **Хранение типа heap**. Этот способ подходит для таблиц и разделов, которые будут получать конкурентные запросы: итеративные UPDATE и DELETE, а также одиночные INSERT. Данный тип лучше всего подходит для таблиц с небольшим объемом строк менее 300 000 записей или менее 100 МБ данных. Данные показатели относительны и зависят от размера кластера.
* **Append-Optimized Storage** – Данные таблицы являются компактными и обладают рядом дополнительных возможностей. Для АО таблиц мы можем выбирать ориентацию строковую или колончатую. Также эти таблицы можно сжимать, применяя различные алгоритмы компрессии. AO таблицы являются менее ресурсоемкими в обслуживании.

В общем случае АО это большие таблицы фактов, которые нечасто обновляются после начальной загрузки, а последующие вставки выполняются только в пакетных операциях.

* **Таблицы, со строчным хранением (Row):** подходят для рабочих нагрузок OLTP-типов с большим количеством итеративных транзакций.
* **Column-Oriented таблицы**: подходят для рабочих нагрузок хранилища данных с агрегацией данных, по небольшому количеству столбцов, или если в таблице есть столбцы, которые требуют регулярных обновлений без изменения других данных столбцов

Констрейнты

Рассмотрим какие виды ограничений существуют в Greenplum

NOT NULL - Колонка с данным ограничением не может иметь значение NULL.

DEFAULT - Если в колонку приходит пустая ячейка, то она заполниться заданным значением по умолчанию

UNIQUE - В колонку с данным ограничением можно вставлять только уникальные значения

PRIMARY KEY - колонка с таким ограничением содержит Уникальные идентификаторы каждой записи в таблице. Данное ограничение задается путем комбинации NOT NULL и UNIQUE, доступно только в HEAP-таблицах.

FOREIGN KEY - Уникальный идентификатор записи в другой таблице допускается в синтаксисе, но не работает

CHECK - Гарантирует, что все значения в колонке соответствуют определённому условию

Партиционирование

Партиционирование Gозволяет разделить большие таблицы, например факты, на логически разделенные куски. Greenplum физически делит части таблиц на сегменте на отдельные таблицы на основании ключа партиционирования.

В отличие от дистрибуции, партиционирование не является необходимым условием для создания таблицы. Это опция, которая используется для оптимизации

Ключ партиционирования – одно поле в таблице, по которому определяется принадлежность записи той или иной партиции (в случае многоуровневого партиционирования – набор полей). В большинстве случаев в качестве ключа партиционирования используется поле с датой.

Грамотное разделение (партиционирование, partitioning) таблиц сокращает объем данных для сканирования, позволяя запросу читать только нужные разделы.

Partition by range позволяет разбивать таблицу по партициям на основе диапазона значений для определенного столбца. Для партиционирования необходимо выбрать столбец, который будет являться ключом партиционировани и указать один или несколько диапазонов партиционирования, которые определяются минимальным и максимальным значением для столбца, выбранного в качестве ключа.

Когда новые данные вставляются в таблицу, Greenplum определяет, к какому диапазону принадлежат данные, на основе ключа партиционирования и определенных вами диапазонов. Это позволяет вам распределять ваши данные по нескольким таблицам. Например, вы можете разбить таблицу продаж по датам, используя PARTITION BY RANGE, чтобы каждая партиция содержала 1 месяц. В таком случае если вы будете использовать в запросе фильтр ограничивая выборку одним месяцом, будет считываться только одна партиция, а не вся таблица целиком.

При партиционировании по списку значений вы разбиваете таблицу на партиции на основе определенных значений в столбце. При использовании PARTITION BY LIST необходимо указывать столбец, содержащий дискретный набор значений, а затем создать разделы на основе этих значений. Каждый раздел содержит строки, в которых значение указанного столбца соответствует определенному значению в списке.

При создании партиций вы можете:

* Указать отличающиеся имена для каждой партиции
* Создать партицию по умолчанию (DEFAULT партишн). В нее попадают значения, которые не входят в диапазоны или списки значений других партиций
* Выбрать разные опции хранения данных в партиции (ориентация, компресия). Например, вы можете разбить партиции по дате и для старых значений, которые редко используются бизнес пользователями указать максимальный уровень компрессии, с целью экономии места на дисек
* Указать отличающиеся диапазоны для каждой партиции. Например для одной таблицы вы можете создать партиции по дню для горячих данных, а холодные данные хранить в партициях по году

При создании партиций невозможно указать разные ключи дистрибуции для партиций одной таблицы. Также нельзя поменять тип партиционирования без пересоздания таблицы.

Источники информации (ссылки)

1. [Официальная документация](https://docs.vmware.com/en/VMware-Greenplum/6/greenplum-database/best_practices-schema.html#storage_model__heap_vs_ao)
2. [Построение DWH на основе Greenplum](https://southbridge.io/blog/tpost/minulc8dx1-postroenie-dwh-na-osnove-greenplum)
3. [10 советов о партиционировании](https://www.bigdataschool.ru/blog/greenplum-partitioning-and-segments-distribution.html).
4. [Констрейнты](https://postgrespro.ru/docs/postgrespro/13/ddl-constraints)