Arenadata™ Database

Версия - v5.6

Рассчет полезной емкости диска

2018 Arenadata™

Оглавление

1 Расчет полезной емкости диска **3**

2 Вычисление размера данных пользователя **4**

3 Метаданные и лог-файлы **5**

Данный документ помогает рассчитать количество данных, которое можно хранить в СУБД Arenadata DB. Также необходимо иметь в виду, что может потребоваться дополнительное пространство для хранения резервных и загрузочных файлов.

Important: Контактная информация службы поддержки – e-mail: info@arenadata.io

Глава 1

Расчет полезной емкости диска

Чтобы узнать, сколько данных может храниться в кластере **Arenadata DB**, необходимо вычислить доступную емкость диска на одном сегменте, а затем умножить на количество сегментов в кластере.

Следует начать с необработанной емкости физических дисков на хосте сегмента, которые доступны для хранилища данных (*RawCapacity*):

$$DiskSize * NumberOfDisks$$

Также необходимо учесть потери из-за форматирования файловой системы (примерно *10%*) и уровень используемого **RAID**. Например, если используется **RAID-10**, расчет следующий:

$$(RawCapacity * 0.9)/2 = FormattedDiskSpace$$

Для обеспечения оптимальной производительности не стоит полностью заполнять диски. Оптимальный уровень загрузки составляет около *70%*. В итоге, необходимо рассчитать доступное место, учитывая оптимальный уровень загрузки, следующим образом:

$$FormattedDiskSpace * 0.7 = UsableDiskSpace$$

После форматирования массивов RAID-дисков и определения максимальной рекомендуемой емкости (*UsableDiskSpace*) необходимо вычислить доступное пространство для пользовательских данных (U). Использование зеркалирования сегментов для избыточности данных приводит к удвоению размера пользовательских данных ($2 * U$).

Архитектура **ADB** также требует, чтобы было зарезервировано место для рабочей области по активным запросам. Рабочее пространство должно составлять примерно одну треть от размера пользовательских данных ($U/3$):

С зеркалами: $(2 * U) + U/3 = UsableDiskSpace$

Без зеркал: $U + U/3 = UsableDiskSpace$

Данные рекомендации предполагают типичную аналитическую нагрузку. Большое число одновременно работающих запросов, а также запросы, требующие большого объема временных файлов, потребуют большего свободного пространства. Как правило, общая пропускная способность системы может быть увеличена путем уменьшения использования рабочего пространства за счет надлежащего управления рабочей нагрузкой. Кроме того, пространство для временного хранения и пространство пользователя могут быть изолированы друг от друга.

Глава 2

Вычисление размера данных пользователя

Как и во всех базах данных, размер исходных данных всегда получается немного больше после их загрузки. В среднем, исходные данные занимают примерно в *1,4* раза больше места после их загрузки (значение коэффициента может отличаться в зависимости от используемых типов данных, типа хранения таблиц, сжатия в базе данных и так далее).

- *Page Overhead* – при загрузке данных в **ADB** они делятся на страницы по *32 КБ*. Каждая страница содержит *20* байта служебных данных страницы;
- *Row Overhead* – для heap-таблиц; каждая строка данных имеет *24* байта служебных данных строки. В append optimized таблице имеется только *4* байта служебных данных для каждой строки;
- *Attribute Overhead* – объём служебных данных для каждого индивидуального значения варьируется в зависимости от выбранного типа данных. Как правило, необходимо использовать наименьший возможный тип данных (при условии, что известны возможные значения, которые имеет столбец);

Индексы распределяются по хостам сегментов, как таблицы. По умолчанию тип индекса в базе данных **ADB** – это B-дерево. Поскольку размер индекса зависит от количества уникальных значений в нем и данных, которые должны быть добавлены, предварительное вычисление точного размера индекса невозможно. Однако, можно приблизительно оценить размер индекса, используя следующие формулы:

B-tree: $UniqueValues * (DataTypeSize + 24)$

Bitmap: $(UniqueValues * NumberOfRows * 1 * CompressionRatio/8) + (UniqueValues * 32)$

Глава 3

Метаданные и лог-файлы

На каждом сегменте хоста также следует учесть пространство для лог-файлов базы данных и метаданных:

- *Метаданные системы* – для каждого инстанса сегмента (*primary* или *mirror*) или инстанса мастера, запущенного на хосте, следует оставить приблизительно *20 МБ* для системных каталогов и метаданных;
- *Write Ahead Log* – для каждого сегмента или мастер-инстанса на хосте, необходимо выделить место для лог-файла *Write-Ahed Log* (*WAL*). *WAL* разделен на сегменты по *64 МБ* каждый. Максимальное количество файлов *WAL* вычисляется по формуле:

$$2 * CheckpointSegments + 1$$

Получившееся значение можно использовать для оценки требований к пространству под *WAL*. По умолчанию параметр *CheckpointSegments* для интсанса равен *8*, то есть *1088 МБ* пространства *WAL*, выделенного для каждого сегмента или мастер-инстанса на хосте.

- *Лог-файлы базы данных ADB* – каждый инстанс сегмента и мастер-инстанс генерируют лог-файлы, которые со временем растут. Для данных лог-файлов должно быть выделено достаточно места. При этом следует использовать механизм ротации для обеспечения того, чтобы лог-файлы не становились слишком большими.