**Task 3 (Cluster)**

Папка задания в репозитории:

<https://github.com/artemgur/AdvancedPostgreSQL/tree/master/Task%203%20(Cluster)>

Flyway-скрипты хранятся в следующей папке:

<https://github.com/artemgur/AdvancedPostgreSQL/tree/master/flyway/sql>

Flyway-скрипты, относящиеся к этому заданию, имеют номер версии вида 3.x.x.

Создание таблицы:

<https://github.com/artemgur/AdvancedPostgreSQL/blob/master/flyway/sql/V3.0__CreateTable.sql>

Таблицу я сделал unlogged. Это значительно ускорит подготовительные операции, но не повлияет на результаты задания, так как наша задача — исследовать время выполнения SELECT.

Вставка строк:

<https://github.com/artemgur/AdvancedPostgreSQL/blob/master/flyway/sql/V3.1.0__Insert.sql>

Перетасовывание строк таблицы:

<https://github.com/artemgur/AdvancedPostgreSQL/blob/master/flyway/sql/V3.1.1__ShuffleUpdate.sql>

После этого был сделан VACUUM ANALYZE таблицы.

<https://github.com/artemgur/AdvancedPostgreSQL/blob/master/flyway/sql/V3.1.2__VacuumAnalyze.sql>

В этот момент запускается PgBench для получения результатов для некластеризованной таблицы.

PgBench для SELECT отдельных значений:

Команда:

<https://github.com/artemgur/AdvancedPostgreSQL/blob/master/Task%203%20(Cluster)/PgBenchSelectSeparate.sh>

Скрипт:

<https://github.com/artemgur/AdvancedPostgreSQL/blob/master/Task%203%20(Cluster)/SelectSeparate.sql>

PgBench для SELECT интервалов:

Команда:

<https://github.com/artemgur/AdvancedPostgreSQL/blob/master/Task%203%20(Cluster)/PgBenchSelectContinuous.sh>

Скрипт:

<https://github.com/artemgur/AdvancedPostgreSQL/blob/master/Task%203%20(Cluster)/SelectContinuous.sql>

id, строки с которыми мы ищем с помощью SELECT, разные на каждой итерации. Чтобы добиться этого я, как и в предыдущем задании, использовал последовательность в качестве переменной-счетчика. Это было сделано, чтобы минимизировать влияние кэширования.

Каждый SELECT получает 40 строк таблицы.

Кластеризация:

<https://github.com/artemgur/AdvancedPostgreSQL/blob/master/flyway/sql/V3.2.0__Cluster.sql>

Analyze:

<https://github.com/artemgur/AdvancedPostgreSQL/blob/master/flyway/sql/V3.2.1__Analyze.sql>

Далее запускаются те же команды PgBench, что ранее, для получения результатов для кластеризованной таблицы.

**Результаты для некластеризованной таблицы:**

SELECT отдельных значений:

* 1.923 ms — latency
* 520.048245 — tps (including connections establishing)
* 520.187205 — tps (excluding connections establishing)

SELECT интервала:

* 0.645 ms — latency
* 1550.618090 — tps (including connections establishing)
* 1551.762477 — tps (excluding connections establishing)

**Результаты для кластеризованной таблицы:**

SELECT отдельных значений:

* 1.873 ms — latency
* 533.883829 — tps (including connections establishing)
* 534.009659 — tps (excluding connections establishing)

SELECT интервала:

* 0.581 ms — latency
* 1719.855299 — tps (including connections establishing)
* 1721.205608 — tps (excluding connections establishing)

**Выводы**

Кластеризация действительно несколько ускоряет время работы SELECT, причем как в случае отдельных значений, нак и в случае интервалов.

Еще видно, что SELECT работает гораздо быстрее в случае интервалов, чем в случае отдельных значений. Это объясняется тем, что в случае отдельных значений нужно искать все значения по отдельности. В то же время, в случае интервала, B-tree дает возможность быстро найти в индексе значения, лежащие между максимальным и минимальным значениями.