Федеральное государственное автономное

образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт Космических и информационных технологий

институт

Кафедра «Информатика»

кафедра

Отчёт о научно-исследовательской работе

Сравнение скорости CRUD операций (Clickhouse c PostgreSQL)

тема

|  |  |  |
| --- | --- | --- |
| Руководитель |  | А. Н. Пупков |
|  | подпись, дата | инициалы, фамилия |
| Студент КИ18/17-1б 031831229 |  | В. А. Прекель |
| номер группы, зачетной книжки | подпись, дата | инициалы, фамилия |

Красноярск 2020

Содержание

[Содержание 2](#_Toc49799774)

[Введение 3](#_Toc49799775)

[1 История создания 4](#_Toc49799776)

[1.1 Разработка IntelliJ IDEA 4](#_Toc49799777)

[Заключение 5](#_Toc49799778)

[Список использованных источников 6](#_Toc49799779)

Индивидуальное задание

Необходимо провести сравнение скорости и анализ CRUD операций для систем управления баз данных Clickhouse и PostgreSQL.

Теоретическая часть:

1. Описать что такое CRUD операции;
2. Описать как работают хранилища данных, ссылаясь на соответствующую документацию (например если вы пишете про ClickHouse, нужно сослаться на документ https://clickhouse.yandex/docs/ru/);
3. Найти информацию о том, как и почему скорость CRUD операций хранилищ отличается, провести сравнительный анализ для каждой операции с детальным и обоснованным объяснением (со ссылками на источники);
4. Сделать выводы о том, почему в данных хранилищах имеются различия в выполнении CRUD операций, чем это вызвано и как дизайн системы влияет на данный параметр.

Экспериментальная часть:

* Установить docker toolbox (или более свежее решение);
* Скачать контейнеры с соответствующими базами данных;
* Написать два простых скрипта выполняющих CRUD операции для каждой из пары баз данных и измеряющих время выполнения;
* Каждый эксперимент провести несколько раз, при этом:
* Нужно указать параметры (виртуальной) машины, на которой проводились исследования (кол-во RAM, CPU, потоков);
* Указать количество итераций для каждого эксперимента;
* Привести значения математического ожидания и дисперсии для каждого результата;
* Сделать графики с пояснениями;
* Сделать выводы о том, почему в данных хранилищах имеются различия в выполнении CRUD операций, чем это вызвано и как дизайн системы влияет на данный параметр.

Введение

Система управления базами данных – набор программного обеспечения, позволяющий определять, обрабатывать, получать и управлять данными в базе данных. [1] Соответственно, основные операции – создания, чтения, изменения и удаления называются CRUD-операциями. В случае с SQL-СУБД, за CRUD операции отвечают соответственно конструкции INSERT, SELECT, UPDATE, DELETE соответственно. [2] В разных системах, с разным дизайном, на идентичной схеме данных, время выполнения CRUD-операций может различаться.

# Теоретическая часть

## Описание СУБД

Существуют строковые и столбцовые (колоночные, column-based) СУБД. Они отличаются тем, что данные, принадлежащие к одной строке в строковых СУБД, хранятся рядом, а в столбцовых СУБД хранятся рядом данные, принадлежащие к одному столбцу. Это позволяет экономить время в запросах на чтение, которые не затрагивают все столбцы таблицы и проводить прочие оптимизации, в том числе по сжатию данных, которые невозможны в столбцовых СУБД. [3] Примеры строковых СУБД: MySQL, PostgreSQL, и MS SQL Server. Примеры столбцовых СУБД: ClickHouse, Vertica, Paraccel, Sybase IQ, Exasol, Infobright, InfiniDB, MonetDB, LucidDB, SAP HANA, Google Dremel, Google PowerDrill, Druid, and kdb+. [4]

ClickHouse – строковая СУБД, предназначенная для анализа данных и OLAP-запросов. Поддерживает SQL, хоть с во многих случаях не совпадающий со стандартом. Представляется как по-настоящему столбцовая СУБД, поддерживающая сжатие данных, хранящая данные на диске, параллельно использующая процессорные ядра, и позволяющая работать на нескольких серверах в кластере. [5] Была разработана в Яндексе и выпущена под открытой лицензией в 2016 году. [6]

PostgreSQL – строковая объектно-реляционная СУБД. Поддерживает большую часть SQL стандарта, а также множество современных функций, такие как сложные запросы, внешние ключи, триггеры, изменяемые представления, транзакционная целостность, многоверсионность. Начинает свою историю из 80-тых годов с проекта POSTGES в Беркли. PostgreSQL основано на последней версии POSTGRES 4.2, выпущенной в 1994 году. [8]

## Анализ работы СУБД

Исходя из того, что PostgreSQL строковая СУБД, то запросы в узких таблицах должны работать быстрее. Так же, ClickHouse предназначен для извлечения и вставки большого количества данных одним запросом, поэтому выполнение множества аналогичных запросов должно быть медленнее, чем в PostgreSQL. [9]

Так же ClickHouse поддерживает операции изменения и удаления через механизм мутаций нестандартным синтаксисом ALTER TABLE … UPDATE и ALTER TABLE … DELETE соответственно. [10] Механизм мутаций не был реализован на релизе Clickhouse и логично предположить, изменение/удаление данных будет работать хуже, чем в строковых СУБД. [11]

# Экспериментальная часть

## Подготовка к эксперименту

Для развёртывания тестируемых СУБД использовался Docker – . Для работы Docker на Windows, использовался Docker Desktop с бэкэндом WSL 2. Были скачаны соответствующие контейнеры yandex/clickhouse-server [] и postgres [].

Написана программа на языке C#, которая подключается к СУБД, выполняет нужные запросы определённое количество раз и сохраняет результаты в CSV-файл. Так как время выполнения запросов может зависеть не только от СУБД, а от программного обеспечения, обеспечивающего подключение клиентского приложения к СУБД, было решено использовать несколько драйверов, использующие технологию ADO.NET. Для ClickHouse: ClickHouse.Ado [], O…Ch []. Для PostgreSQL: Тзпыцд [], dotConnect Express [].

Было решено тестировать на таблице с примитивными данными. В таблице 2 столбца, первый – первичный ключ, второй – значение. Оба столбца – целое 32 битное число, оба столбца не могут принимать значение null.

Листинг 1 – Создание таблицы в ClickHouse и PostgreSQL

…

Для create-операции использован следующий INSERT-запрос, вставляющий запись с ключом i и значением i. Этот запрос повторяется N раз, значение i последовательно растёт от 0 до N-1. Этот запрос идентичен для ClickHouse и PostgreSQL.

Листинг 2 -- …

…

Для read-операции написан SELECT-запрос, который так же N раз вызывается и запрашивает значение с ключём i. Этот запрос идентичен для ClickHouse и PostgreSQL.

Листинг – в

…

Для update-операции значение записи с ключём i заменяется на N-i следующим UPDATE-запросом. Для ClickHouse используется нестандартная конструкция ALTER TABLE … UPDATE.

Листинг – фыв

…

Для delete-операции удаляются записи с 0 по N-1 следующим DELETE-запросом. Для ClickHouse используется нестандартная конструкция ALTER TABLE … DELETE.

Листинг – в

…

Полный код программы указан в приложении А.

Параметры машины: процессор Ryzen 3 1200 3.59 GHz, 4 потока, 16 Gb RAM, SSD.

## Результаты эксперимента

Были проведены замеры для 100, 1000 и 5000 записей.

Таблица – средние значения для 100

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| DbmsAndDriverName | avg(Create) | avg(Read) | avg(Update) | avg(Delete) |
| ClickHouse (ClickHouse.Ado) | 1218,75 | 821,34 | 1402,14 | 1475,33 |
| ClickHouse (Octonica.ClickHouseClient) | 515,61 | 530,12 | 1176,07 | 1236,83 |
| Postgres (Npgsql) | 316,23 | 123,70 | 313,01 | 310,82 |
| Postgres (dotConnect.Express.for.PostgreSQL) | 440,13 | 234,82 | 416,92 | 422,21 |

Таблица – средние значения для 1000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| DbmsAndDriverName | avg(Create) | avg(Read) | avg(Update) | avg(Delete) |
| ClickHouse (ClickHouse.Ado) | 12466,90 | 8617,71 | 14845,54 | 17636,25 |
| ClickHouse (Octonica.ClickHouseClient) | 5164,08 | 5825,96 | 13580,42 | 15463,70 |
| Postgres (Npgsql) | 2944,41 | 1249,75 | 2868,14 | 3067,58 |
| Postgres (dotConnect.Express.for.PostgreSQL) | 4141,51 | 2357,73 | 4154,59 | 4180,64 |

Таблица – средние значения для 5000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| DbmsAndDriverName | avg(Create) | avg(Read) | avg(Update) | avg(Delete) |
| ClickHouse (ClickHouse.Ado) | 59829,90 | 43998,25 | 87143,45 | 96201,52 |
| ClickHouse (Octonica.ClickHouseClient) | 26856,12 | 25999,40 | 79564,14 | 85883,76 |
| Postgres (Npgsql) | 15949,27 | 6240,25 | 14199,52 | 15217,90 |
| Postgres (dotConnect.Express.for.PostgreSQL) | 21423,44 | 12294,57 | 20163,62 | 20084,61 |

Таблица – среднеквадратичное отклонение значений для

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DbmsAndDriverName | RecordCount | stddevPopStable(Create) | stddevPopStable(Read) | stddevPopStable(Update) | stddevPopStable(Delete) |
| ClickHouse (ClickHouse.Ado) | 100 | 436,65 | 426,62 | 778,57 | 752,08 |
| ClickHouse (ClickHouse.Ado) | 1000 | 1845,48 | 1334,42 | 3601,66 | 5672,27 |
| ClickHouse (ClickHouse.Ado) | 5000 | 4684,36 | 4486,35 | 14106,16 | 15475,30 |
| ClickHouse (Octonica.ClickHouseClient) | 100 | 281,57 | 142,02 | 717,51 | 780,70 |
| ClickHouse (Octonica.ClickHouseClient) | 1000 | 1484,75 | 1721,69 | 5942,46 | 6245,25 |
| ClickHouse (Octonica.ClickHouseClient) | 5000 | 3133,14 | 3051,76 | 5524,54 | 5182,94 |
| Postgres (Npgsql) | 100 | 259,76 | 20,05 | 235,76 | 256,03 |
| Postgres (Npgsql) | 1000 | 1280,43 | 198,89 | 1170,70 | 1681,38 |
| Postgres (Npgsql) | 5000 | 4433,23 | 776,54 | 3148,96 | 4561,54 |
| Postgres (dotConnect.Express.for.PostgreSQL) | 100 | 524,68 | 44,31 | 223,69 | 300,89 |
| Postgres (dotConnect.Express.for.PostgreSQL) | 1000 | 1266,77 | 243,26 | 1129,42 | 1183,94 |
| Postgres (dotConnect.Express.for.PostgreSQL) | 5000 | 4124,73 | 2758,63 | 4591,16 | 4345,34 |

Таблица – дисперсия

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DbmsAndDriverName | RecordCount | varPopStable(Create) | varPopStable(Read) | varPopStable(Update) | varPopStable(Delete) |
| ClickHouse (ClickHouse.Ado) | 100 | 190663,47 | 182007,28 | 606167,06 | 565617,16 |
| ClickHouse (ClickHouse.Ado) | 1000 | 3405813,86 | 1780677,27 | 12971936,85 | 32174603,80 |
| ClickHouse (ClickHouse.Ado) | 5000 | 21943247,58 | 20127380,21 | 198983724,94 | 239484965,83 |
| ClickHouse (Octonica.ClickHouseClient) | 100 | 79282,95 | 20170,40 | 514821,26 | 609492,88 |
| ClickHouse (Octonica.ClickHouseClient) | 1000 | 2204468,98 | 2964213,95 | 35312827,37 | 39003140,99 |
| ClickHouse (Octonica.ClickHouseClient) | 5000 | 9816589,63 | 9313224,15 | 30520517,71 | 26862823,78 |
| Postgres (Npgsql) | 100 | 67474,23 | 402,15 | 55582,05 | 65551,42 |
| Postgres (Npgsql) | 1000 | 1639496,44 | 39557,04 | 1370547,54 | 2827024,48 |
| Postgres (Npgsql) | 5000 | 19653503,22 | 603016,45 | 9915934,56 | 20807675,77 |
| Postgres (dotConnect.Express.for.PostgreSQL) | 100 | 275285,86 | 1963,63 | 50035,57 | 90535,59 |
| Postgres (dotConnect.Express.for.PostgreSQL) | 1000 | 1604718,05 | 59173,97 | 1275583,73 | 1401713,74 |
| Postgres (dotConnect.Express.for.PostgreSQL) | 5000 | 17013380,89 | 7610043,46 | 21078783,44 | 18881943,99 |

Заключение

В заключение можно отметить, что благодаря грамотной политики, маркетингу и умелой разработке [16] компания JetBrains удачно конкурирует среди производителей IDE, даже таких как Microsoft, выпустив IDE для .NET Rider.

Список использованных источников

1. <https://www.techopedia.com/definition/24361/database-management-systems-dbms>

1. <https://en.wikipedia.org/wiki/Create,_read,_update_and_delete>

1. <https://en.wikipedia.org/wiki/Column-oriented_DBMS>

1. <https://clickhouse.tech/docs/en/>

1. <https://clickhouse.tech/docs/en/introduction/distinctive-features/>

1. <https://habr.com/en/company/yandex/blog/303282/>

1. <https://www.postgresql.org/docs/12/intro-whatis.html>

1. <https://www.postgresql.org/docs/12/history.html>

1. <https://clickhouse.tech/docs/en/introduction/performance/>

1. <https://clickhouse.tech/docs/en/sql-reference/statements/alter/>

1. <https://clickhouse.tech/blog/en/2016/how-to-update-data-in-clickhouse/>