HBase

HBase là một cơ sở dữ liệu phân tán không quan hệ mã nguồn mở được mô phỏng theo Bigtable của Google và được viết bằng Java. Nó được phát triển như một phần của dự án Apache Hadoop của Apache Software Foundation và chạy trên HDFS hoặc Alluxio, cung cấp các khả năng giống như Bigtable cho Hadoop.

Giới thiệu về HBase

HBase là một cơ sở dữ liệu phân tán mã nguồn mở xây dựng dựa trên kiến trúc Hadoop. Nó được thiết kế để xử lý dữ liệu lớn và có khả năng mở rộng cao, giúp các tổ chức và doanh nghiệp lưu trữ, quản lý và truy xuất dữ liệu một cách hiệu quả. Trong bài viết này, chúng ta sẽ khám phá chi tiết về HBase, từ đặc điểm, cấu trúc đến tính năng và ứng dụng của nó.

1. Đặc điểm của HBase:

- Mở rộng: HBase được xây dựng trên Hadoop Distributed File System (HDFS), cho phép mở rộng lưu trữ và xử lý dữ liệu trên nhiều máy chủ.
- Phân tán: Dữ liệu trong HBase được phân tán trên các nút máy chủ và được sao chép để đảm bảo tính sẵn sàng và bảo mật.
- Hỗ trợ dữ liệu cấu trúc: HBase hỗ trợ lưu trữ dữ liệu cấu trúc, cho phép tạo các bảng với hàng và cột. Điều này giúp tăng tốc độ truy xuất dữ liệu và cung cấp khả năng tìm kiếm nhanh chóng.
- Cung cấp tính nhất quán: HBase sử dụng hệ thống ghi nhật ký (write-ahead log) để đảm bảo tính nhất quán của dữ liệu và khả năng khôi phục sau sự cố.

2. Cấu trúc của HBase:

HBase sử dụng một cấu trúc bảng dữ liệu dựa trên hàng và cột. Bảng được chia thành các hàng (rows) và mỗi hàng chứa nhiều cột (columns). Các hàng được xác định bằng một khóa chính (primary key), trong khi các cột được xác định bằng tên cột (column name). Mỗi cột chứa một giá trị (value) và một nhãn thời gian (timestamp) cho phép lưu trữ nhiều phiên bản của cùng một giá trị.

3. Tính năng của HBase:

- Đọc/ghi dữ liệu nhanh chóng: HBase được tối ưu hóa cho việc đọc và ghi dữ liệu lớn với hiệu suất cao. Dữ liệu được lưu trữ trong bộ nhớ đệm (in-memory cache) để tăng tốc độ truy xuất.
- Khả năng mở rộng: HBase có khả năng mở rộng tuyến tính, cho phép thêm nút máy chủ để tăng khả năng lưu trữ và xử lý dữ liệu.
- Hỗ trợ tìm kiếm và truy vấn: HBase cung cấp các công cụ truy vấn linh hoạt và mạnh mẽ, bao gồm truy vấn theo khóa chính, quét toàn bộ bảng và truy vấn phức tạp sử dụng

Apache HBase Filters.
- Sao lưu và phục hồi: HBase cung cấp khả năng sao lưu và khôi phục dữ liệu, giúp đảm bảo tính sẵn sàng và bảo mật.

4. Ứng dụng của HBase:

- HBase được sử dụng trong các ứng dụng lưu trữ dữ liệu lớn như hệ thống xử lý log, hệ thống thu thập dữ liệu thời gian thực và hệ thống phân tích dữ liệu.
- Nó cũng được sử dụng trong các hệ thống tìm kiếm, hệ thống xác thực người dùng và hệ thống theo dõi sự kiện.
- HBase cũng được ứng dụng trong lĩnh vực trò chơi điện tử, quảng cáo trực tuyến và phân tích dữ liệu thương mại điện tử.

Tóm lại, HBase là một cơ sở dữ liệu phân tán và mở rộng được thiết kế để xử lý dữ liệu lớn. Với khả năng mở rộng, tính nhất quán và hiệu suất cao, HBase đã trở thành một công cụ quan trọng cho việc lưu trữ và truy xuất dữ liệu trong các hệ thống xử lý dữ liệu lớn và ứng dụng có yêu cầu cao về hiệu suất.

So sánh HBase và RDBMS

HBase và RDBMS (Relational Database Management System) là hai loại cơ sở dữ liệu có những đặc điểm và ứng dụng khác nhau. Dưới đây là một so sánh chi tiết giữa HBase và RDBMS từ nhiều khía cạnh khác nhau:

1. Cấu trúc dữ liệu

- RDBMS: RDBMS sử dụng mô hình dữ liệu có cấu trúc, dựa trên bảng, hàng và cột. Dữ liệu được tổ chức thành các bảng với các quan hệ (relationships) giữa chúng thông qua các khóa ngoại.
- HBase: HBase sử dụng mô hình dữ liệu không có cấu trúc, dựa trên các bảng cột (columnar). Dữ liệu được lưu trữ dưới dạng hàng và cột, với khóa chính duy nhất xác định hàng.

2. Khả năng mở rộng

- RDBMS: RDBMS không dễ dàng mở rộng theo quy mô lớn. Việc thêm nút máy chủ mới có thể gây ra các vấn đề về hiệu suất và khả năng phân tán.
- HBase: HBase được thiết kế để mở rộng dựa trên kiến trúc phân tán. Nó có khả năng linh hoạt trong việc thêm nút máy chủ mới để tăng cường khả năng lưu trữ và xử lý dữ liệu.

3. Hiệu suất

- RDBMS: RDBMS thường có hiệu suất cao trong các truy vấn phức tạp và thao tác kết nối nhiều bảng. Các truy vấn SQL có thể được tối ưu hóa để đạt hiệu suất tốt.
- HBase: HBase thường có hiệu suất cao trong việc đọc và ghi dữ liệu lớn. Nó được tối ưu hóa cho việc xử lý dữ liệu trên quy mô lớn và có khả năng truy vấn nhanh chóng trên dữ liệu cấu trúc không đồng nhất.

4. Tính nhất quán dữ liệu

- RDBMS: RDBMS đảm bảo tính nhất quán dữ liệu thông qua các ràng buộc toàn vẹn (integrity constraints) như khóa ngoại và quy tắc kiểm tra (check constraints).
- HBase: HBase không cung cấp tính nhất quán dữ liệu tự động như RDBMS. Tuy nhiên, nó hỗ trợ các tính năng như hệ thống ghi nhật ký (write-ahead log) để đảm bảo sự nhất quán

trong trường hợp sự cố.

5. Ứng dụng

- RDBMS: RDBMS thường được sử dụng trong các ứng dụng yêu cầu tính toán phức tạp và truy vấn đa dạng như hệ thống quản lý thông tin khách hàng (CRM), hệ thống quản lý cơ sở dữ liệu (DBMS) và hệ thống quản lý dữ liệu doanh nghiệp (ERP).
- HBase: HBase thích hợp cho các ứng dụng có yêu cầu về việc lưu trữ và xử lý dữ liệu lớn, như hệ thống xử lý log, hệ thống theo dõi thời gian thực và phân tích dữ liệu trên quy mô lớn.

6. Tính linh hoạt

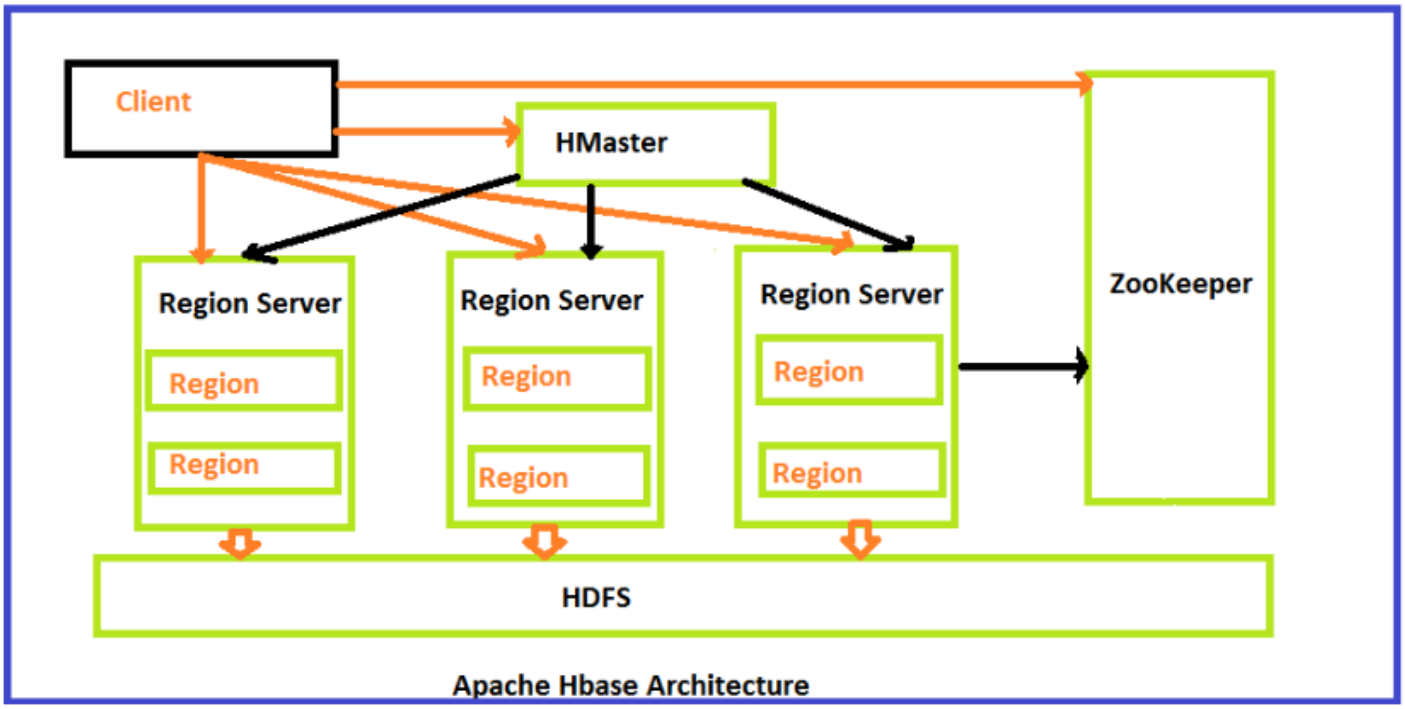
- RDBMS có tính linh hoạt (flexibility) thấp hơn so với HBase, vì cấu trúc dữ liệu quan hệ không thể thay đổi dễ dàng. Việc thay đổi cấu trúc dữ liệu thường đòi hỏi các thao tác phức tạp như cập nhật (update) hoặc thay đổi kiểu dữ liệu.
- HBase có tính linh hoạt cao hơn so với RDBMS, vì cấu trúc dữ liệu có thể thay đổi dễ dàng. Việc thêm hoặc xóa các cột hay hàng trong HBase đều được thực hiện một cách linh hoạt.

Tóm lại, RDBMS và HBase có những đặc điểm và ứng dụng riêng biệt. RDBMS thích hợp cho các ứng dụng yêu cầu tính toán phức tạp và truy vấn đa dạng, trong khi HBase là lựa chọn tốt cho việc xử lý dữ liệu lớn và việc truy xuất nhanh chóng trên quy mô lớn.

HBase được thiết kế để xử lý các tập dữ liệu lớn và có khả năng mở rộng tuyến tính, trong khi RDBMS không được thiết kế để xử lý các tập dữ liệu lớn và không có khả năng mở rộng tuyến tính. RDBMS có tính nhất quán cao hơn so với HBase, trong khi HBase có tính linh hoạt cao hơn so với RDBMS.

Kiến trúc của HBase

HBase là một hệ thống cơ sở dữ liệu phân tán và có khả năng mở rộng tuyến tính. HBase được thiết kế để lưu trữ và xử lý các tập dữ liệu lớn.



Các thành phần chính của HBase bao gồm:

1. HMaster

HMaster là thành phần quản lý cụm HBase. HMaster quản lý các bảng (table), các cụm máy tính (region server), các định vị (location) của các cột và các hàng trong bảng, và các thay đổi trong cấu trúc của bảng. HMaster cũng quản lý các yêu cầu của khách hàng (client request) và phân phối chúng đến các region server tương ứng.

2. Region Server

Region Server chịu trách nhiệm lưu trữ và xử lý các dữ liệu của một hoặc nhiều bảng trong HBase. Region Server thường được triển khai trên các cụm máy tính riêng biệt trong mạng. Mỗi Region Server quản lý một hoặc nhiều phân vùng (region) của các bảng trong HBase. Các phân vùng được phân phối đồng đều trên các Region Server để đảm bảo tính cân bằng tải trong hệ thống.

3. ZooKeeper

ZooKeeper là một dịch vụ phân tán chịu trách nhiệm quản lý thông tin cấu hình và trạng thái của các thành phần trong HBase. ZooKeeper cung cấp các tính năng như cơ chế khóa (locking), đồng

bộ hoá (synchronization), và phân phối (distribution) để đảm bảo tính nhất quán và độ tin cậy của hệ thống.

4. HDFS

HDFS (Hadoop Distributed File System) là hệ thống lưu trữ phân tán được sử dụng để lưu trữ dữ liệu của HBase. HDFS cung cấp tính năng lưu trữ dữ liệu trên nhiều nút máy tính trong mạng, giúp đảm bảo tính bền vững và khả năng mở rộng của hệ thống.

5. HBase Client

HBase Client là thành phần cung cấp API để khách hàng có thể truy xuất và thay đổi dữ liệu trong HBase. HBase Client cung cấp các phương thức để thực hiện các thao tác như tìm kiếm dữ liệu, chèn dữ liệu, cập nhật dữ liệu, xóa dữ liệu, và quản lý cấu trúc của bảng.

6. Tổng kết

Trên đây là một số thành phần chính của kiến trúc của HBase. HBase được thiết kế để lưu trữ và xử lý các tập dữ liệu lớn, và có khả năng mở rộng tuyến tính. HBase cũng hỗ trợ các tính năng như tính nhất quán cao, tính sẵn sàng cao, tính linh hoạt cao, và tính bảo mật cao.

Hướng dẫn cài đặt HBase trên Centos

Để cài đặt HBase trên CentOS, bạn cần làm theo các bước sau:

- Cài đặt Java:

```
sudo yum install java-1.8.0-openjdk
```

- Tải HBase từ trang web chính thức:

```
wget <https://www.apache.org/dist/hbase/2.4.7/hbase-2.4.7-bin.tar.gz>
```

- Giải nén tệp tin HBase:

```
tar -xvzf hbase-2.4.7-bin.tar.gz
```

- Di chuyển thư mục HBase:

```
cd hbase-2.4.7/
```

- Cấu hình tệp tin `hbase-env.sh`:

```
vi conf/hbase-env.sh
```

Thêm đoạn mã sau vào cuối tệp tin:

```
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk
export HBASE_MANAGES_ZK=false
```

- Chỉnh sửa tệp tin `hbase-site.xml`:

```
vi conf/hbase-site.xml
```

Thêm đoạn mã sau vào cuối tệp tin:

```
<configuration>
  <property>
    <name>hbase.rootdir</name>
    <value>file:///home/hbase-2.4.7/data</value>
  </property>
  <property>
    <name>hbase.zookeeper.property.dataDir</name>
    <value>/home/hbase-2.4.7/zookeeper</value>
  </property>
  <property>
    <name>hbase.unsafe.stream.capability.enforce</name>
    <value>false</value>
    <description>
      Controls whether HBase will check for stream capabilities (hflush/hsync).
      Disable this if you intend to run on LocalFileSystem, denoted by a rootdir
      with the 'file://' scheme, but be mindful of the NOTE below.
    </description>
  </property>
</configuration>
```

- Tạo thư mục để lưu trữ dữ liệu và khởi động HBase:

```
mkdir data
./bin/start-hbase.sh
```

- Kiểm tra xem HBase đã hoạt động chưa:

```
jps
```

Nếu bạn nhìn thấy `HMaster` và `HRegionServer` trong kết quả, có nghĩa là HBase đã được cài đặt thành công.

HBase Shell

HBase cung cấp một giao diện dòng lệnh (HBase shell) cho phép bạn truy cập và thao tác với dữ liệu trong HBase. HBase shell là một công cụ mạnh mẽ cho phép bạn thực hiện các thao tác như tạo bảng, thêm dữ liệu vào bảng, xóa dữ liệu khỏi bảng, và truy vấn dữ liệu từ bảng.

Để bắt đầu sử dụng HBase shell, bạn cần mở cửa sổ dòng lệnh và nhập lệnh sau:

```
hbase shell
```

Sau đó, bạn có thể sử dụng các lệnh HBase shell để thao tác với dữ liệu trong HBase. Các lệnh này bao gồm:

- `create` để tạo bảng mới
- `describe` để mô tả các thuộc tính của bảng
- `put` để thêm dữ liệu vào bảng
- `get` để truy vấn dữ liệu từ bảng
- `scan` để duyệt dữ liệu trong bảng
- `delete` để xóa dữ liệu khỏi bảng
- `disable` để tạm dừng bảng
- `enable` để kích hoạt lại bảng
- `drop` để xóa bảng

Ví dụ, để tạo một bảng mới với tên là `mytable` và hai cột là `col1` và `col2`, bạn có thể nhập lệnh sau:

```
create 'mytable', 'col1', 'col2'
```

Sau đó, bạn có thể sử dụng lệnh `put` để thêm dữ liệu vào bảng `mytable`:

```
put 'mytable', 'row1', 'col1', 'value1'
put 'mytable', 'row1', 'col2', 'value2'
```

Lệnh này sẽ thêm hai giá trị `value1` và `value2` vào hàng `row1` của bảng `mytable`, trong hai cột `col1` và `col2`.

HBase Shell là một công cụ mạnh mẽ để quản lý và tương tác với dữ liệu trong HBase. Nó cung cấp các chức năng cơ bản để tạo, đọc, ghi và xóa dữ liệu trong HBase từ giao diện dòng lệnh dễ sử dụng.

HBase - Create table

Trong HBase, để tạo một bảng mới, bạn sử dụng lệnh `create`. Bảng trong HBase được định nghĩa bởi tên bảng và danh sách các cột. Các cột trong HBase được xác định bởi tên cột và các thuộc tính của cột đó.

Ví dụ, để tạo bảng mới với tên là `mytable` và hai cột là `col1` và `col2`, bạn có thể sử dụng lệnh sau:

```
create 'mytable', 'col1', 'col2'
```

Khi bạn tạo một bảng mới, HBase sẽ tự động tạo một số cột thông tin, bao gồm `rowkey`, `column family`, và `column qualifier`. Dữ liệu trong HBase được tổ chức thành các hàng, mỗi hàng có một `rowkey` duy nhất để xác định hàng đó. Mỗi hàng có thể chứa nhiều cột, được phân loại thành các `column family`. Mỗi `column family` có thể chứa nhiều `column qualifier`, là các cột còn lại.

Sau khi bạn đã tạo bảng trong HBase, bạn có thể sử dụng lệnh `put` để thêm dữ liệu vào bảng. Ví dụ, để thêm một giá trị vào hàng `row1` của bảng `mytable`, trong cột `col1`, bạn có thể sử dụng lệnh sau:

```
put 'mytable', 'row1', 'col1', 'value1'
```

Lệnh này sẽ thêm giá trị `value1` vào hàng `row1` của bảng `mytable`, trong cột `col1`. Bạn có thể thêm nhiều giá trị vào cùng một hàng bằng cách sử dụng nhiều lệnh `put` khác nhau.

Ngoài ra, bạn cũng có thể sử dụng lệnh `scan` để truy vấn và lấy dữ liệu từ bảng. Lệnh này cho phép bạn duyệt qua tất cả các hàng trong bảng và lấy dữ liệu từ mỗi cột. Ví dụ, để lấy tất cả dữ liệu từ bảng `mytable`, bạn có thể sử dụng lệnh sau:

```
scan 'mytable'
```

Lệnh này sẽ trả về tất cả dữ liệu trong bảng `mytable`. Bạn có thể sử dụng các tham số khác của lệnh `scan` để lọc và sắp xếp dữ liệu theo các tiêu chí khác nhau.

Đó là một số khái niệm cơ bản về cách tạo bảng và thêm dữ liệu vào bảng trong HBase. HBase cung cấp nhiều lệnh và tính năng khác để quản lý và truy vấn dữ liệu, tùy thuộc vào nhu cầu của

bạn.

HBase - Listing tables

Để liệt kê các bảng hiện có trong HBase, bạn có thể sử dụng HBase Shell và thực hiện các bước sau:

- Khởi động HBase Shell: Mở terminal và chạy lệnh sau:

```
hbase shell
```

- Liệt kê các bảng: Sử dụng lệnh `list` để hiển thị danh sách các bảng có sẵn trong HBase:

```
list
```

HBase Shell sẽ hiển thị danh sách các bảng, cùng với các thông tin khác như tên bảng, số lượng gia đình cột (column family) và các thông số khác.

- Ví dụ:

- Khi bạn chạy lệnh `list`, HBase Shell sẽ trả về kết quả tương tự như sau:

```
TABLE
my_table
another_table
2 row(s)
```

- Trong ví dụ trên, HBase Shell đang hiển thị hai bảng: "my_table" và "another_table". Dòng cuối cùng cho biết tổng số bảng (2 bảng) và tổng số hàng (2 hàng) trong HBase.

Lưu ý: Khi sử dụng HBase Shell, hãy đảm bảo đã khởi động HBase và đang chạy trong môi trường Shell để có thể thực hiện các lệnh tương tác với HBase.

HBase - Disabling table

Để tạm dừng một bảng trong HBase, bạn có thể sử dụng lệnh `disable` trong HBase shell. Lệnh này cho phép bạn tạm dừng bảng và ngăn chặn các hoạt động ghi mới vào bảng. Tuy nhiên, dữ liệu trong bảng vẫn được lưu trữ và có thể được truy cập.

Để sử dụng lệnh `disable`, bạn chỉ cần nhập lệnh sau trong HBase shell:

```
disable 'ten_bang'
```

Trong đó, `ten_bang` là tên của bảng mà bạn muốn tạm dừng. Lệnh này sẽ ngay lập tức tạm dừng bảng và ngăn chặn các hoạt động ghi mới vào bảng.

Khi bảng đã bị tạm dừng, bạn có thể thực hiện một số thao tác khác như sửa đổi cấu trúc của bảng hoặc sao chép dữ liệu từ bảng này sang bảng khác.

HBase - Enabling table

Để kích hoạt lại một bảng trong HBase, bạn có thể sử dụng lệnh `enable` trong HBase shell. Lệnh này cho phép bạn kích hoạt lại bảng và cho phép các hoạt động ghi mới vào bảng.

Để sử dụng lệnh `enable`, bạn chỉ cần nhập lệnh sau trong HBase shell:

```
enable 'ten_bang'
```

Trong đó, `ten_bang` là tên của bảng mà bạn muốn kích hoạt lại. Lệnh này sẽ kích hoạt lại bảng và cho phép các hoạt động ghi mới vào bảng.

HBase - Describe table

Để mô tả các thuộc tính của một bảng trong HBase, bạn có thể sử dụng lệnh `describe` trong HBase shell. Lệnh này sẽ hiển thị thông tin về các cột trong bảng, cũng như các thuộc tính khác của bảng.

Để sử dụng lệnh `describe`, bạn chỉ cần nhập lệnh sau trong HBase shell:

```
describe 'tên_bảng'
```

Trong đó, `'tên_bảng'` là tên của bảng mà bạn muốn mô tả. Lệnh này sẽ hiển thị thông tin chi tiết về bảng, bao gồm các cột và các thuộc tính khác của bảng.

Đó là cách sử dụng lệnh `describe` để mô tả các thuộc tính của một bảng trong HBase.

HBase - Alter table

Để sửa đổi cấu trúc của một bảng trong HBase, bạn có thể sử dụng lệnh `alter` trong HBase shell. Lệnh này cho phép bạn thêm hoặc xóa các cột, thay đổi tên bảng, thêm hoặc xóa các thuộc tính của bảng, và nhiều hơn nữa.

Để sử dụng lệnh `alter`, bạn chỉ cần nhập lệnh sau trong HBase shell:

```
alter 'ten_bang', {OPTIONS}
```

Trong đó, `ten_bang` là tên của bảng mà bạn muốn sửa đổi, và `{OPTIONS}` là một danh sách các tuỳ chọn để thực hiện các thay đổi cần thiết trên bảng.

Bạn có thể sử dụng các tuỳ chọn sau để sửa đổi bảng trong HBase:

- `NAME => 'new_table_name'`: Đổi tên của bảng thành `new_table_name`.
- `COMPRESSION => 'NONE'`: Vô hiệu hóa nén dữ liệu cho bảng.
- `COMPRESSION => 'SNAPPY'`: Sử dụng nén Snappy cho dữ liệu bảng.
- `COMPRESSION => 'LZO'`: Sử dụng nén LZO cho dữ liệu bảng.
- `COMPRESSION => 'GZ'`: Sử dụng nén Gzip cho dữ liệu bảng.
- `ADD => {NAME => 'new_column', VERSIONS => 1, COMPRESSION => 'NONE', TTL => 2592000, BLOCKSIZE => 65536}`: Thêm một cột mới vào bảng với tên là `new_column`, một phiên bản, không nén dữ liệu, một thời gian sống (TTL) là 30 ngày, và một kích thước khối là 64 KB.
- `DROP => 'column_name'`: Xóa cột có tên là `column_name` khỏi bảng.
- `MAX_FILESIZE => '10000000'`: Thiết lập kích thước tối đa của một tệp dữ liệu là 10 MB.
- `MEMSTORE_FLUSHSIZE => '52428800'`: Thiết lập kích thước tối đa của bộ đệm memstore là 50 MB.

Ví dụ, để thêm một cột mới vào bảng `mytable` với tên là `new_column`, bạn có thể nhập lệnh sau:

```
alter 'mytable', {NAME => 'new_column', VERSIONS => 1}
```

Lệnh này sẽ thêm một cột mới với tên là `new_column` vào bảng `mytable`, với một phiên bản.

Đó là cách sử dụng lệnh `alter` để sửa đổi cấu trúc của một bảng trong HBase.

HBase - Drop table

Để xóa một bảng trong HBase, bạn có thể sử dụng lệnh `disable` để tạm dừng bảng trước khi xóa. Sau đó, bạn có thể sử dụng lệnh `drop` để xóa bảng.

Để tạm dừng một bảng trong HBase, bạn có thể sử dụng lệnh `disable` trong HBase shell. Lệnh này cho phép bạn tạm dừng bảng và ngăn chặn các hoạt động ghi mới vào bảng. Tuy nhiên, dữ liệu trong bảng vẫn được lưu trữ và có thể được truy cập.

Để sử dụng lệnh `disable`, bạn chỉ cần nhập lệnh sau trong HBase shell:

```
disable 'ten_bang'
```

Trong đó, `ten_bang` là tên của bảng mà bạn muốn tạm dừng. Lệnh này sẽ ngay lập tức tạm dừng bảng và ngăn chặn các hoạt động ghi mới vào bảng.

Sau khi bảng đã bị tạm dừng, bạn có thể sử dụng lệnh `drop` để xóa bảng. Để sử dụng lệnh `drop`, bạn chỉ cần nhập lệnh sau trong HBase shell:

```
drop 'ten_bang'
```

Trong đó, `ten_bang` là tên của bảng mà bạn muốn xóa. Lệnh này sẽ xóa bảng và các dữ liệu liên quan khỏi HBase.

Đó là cách sử dụng lệnh `disable` và `drop` để tạm dừng và xóa bảng trong HBase.

HBase - Put data

Để thêm dữ liệu vào bảng trong HBase, bạn có thể sử dụng lệnh `put`. Lệnh `put` cho phép bạn thêm một giá trị mới vào một hàng cụ thể trong bảng của bạn. Để sử dụng lệnh `put`, bạn cần chỉ định tên bảng, tên hàng, tên cột và giá trị bạn muốn thêm vào.

Cú pháp của lệnh `put` như sau:

```
put 'table_name', 'row_key', 'column_family:column_name', 'value'
```

Trong đó:

- `table_name` là tên của bảng bạn muốn thêm dữ liệu vào.
- `row_key` là khóa của hàng bạn muốn thêm dữ liệu vào.
- `column_family` là tên của gia đình cột chứa cột bạn muốn thêm.
- `column_name` là tên của cột bạn muốn thêm.
- `value` là giá trị bạn muốn thêm.

Ví dụ, để thêm một giá trị `John` vào hàng `1` của bảng `user_table` trong cột `info:name`, bạn có thể sử dụng lệnh sau:

```
put 'user_table', '1', 'info:name', 'John'
```

Lệnh này sẽ thêm giá trị `John` vào hàng `1` của bảng `user_table` trong cột `info:name`.

Bạn cũng có thể thêm nhiều giá trị cho cùng một hàng bằng cách sử dụng nhiều lệnh `put` khác nhau. Ví dụ, để thêm giá trị `30` vào cột `info:age` của hàng `1` trong bảng `user_table`, bạn có thể sử dụng lệnh sau:

```
put 'user_table', '1', 'info:age', '30'
```

Lệnh này sẽ thêm giá trị `30` vào cột `info:age` của hàng `1` trong bảng `user_table`.

Với lệnh `put`, bạn có thể thêm bất kỳ giá trị nào cho bất kỳ hàng hoặc cột nào trong bảng của bạn. Điều này giúp bạn tạo ra một bảng linh hoạt và phù hợp với nhu cầu của bạn.

HBase - Update data

Để cập nhật dữ liệu trong bảng HBase, bạn có thể sử dụng lệnh `put`. Lệnh `put` cho phép bạn cập nhật một giá trị mới vào một hàng cụ thể trong bảng của bạn.

Để sử dụng lệnh `put`, bạn cần chỉ định tên bảng, tên hàng, tên cột và giá trị bạn muốn cập nhật.

Cú pháp của lệnh `put` như sau:

```
put 'ten_bang', 'hang_key', 'ten_cot', 'gia_tri'
```

Trong đó:

- `ten_bang` là tên của bảng bạn muốn cập nhật dữ liệu.
- `hang_key` là khóa của hàng bạn muốn cập nhật dữ liệu.
- `ten_cot` là tên của cột bạn muốn cập nhật.
- `gia_tri` là giá trị bạn muốn cập nhật.

Ví dụ, để cập nhật giá trị của cột `age` thành `35` cho hàng `1` trong bảng `user_table`, bạn có thể sử dụng lệnh sau:

```
put 'user_table', '1', 'info:age', '35'
```

Lệnh này sẽ cập nhật giá trị của cột `age` thành `35` cho hàng `1` trong bảng `user_table`.

Nếu bạn muốn cập nhật nhiều giá trị cho cùng một hàng, bạn có thể sử dụng nhiều lệnh `put` khác nhau. Ví dụ, để cập nhật giá trị của cột `age` thành `35` và giá trị của cột `name` thành `John` cho hàng `1` trong bảng `user_table`, bạn có thể sử dụng lệnh sau:

```
put 'user_table', '1', 'info:age', '35'
put 'user_table', '1', 'info:name', 'John'
```

Lệnh này sẽ cập nhật giá trị của cột `age` thành `35` và giá trị của cột `name` thành `John` cho hàng `1` trong bảng `user_table`.

Với lệnh `put`, bạn có thể cập nhật bất kỳ giá trị nào cho bất kỳ hàng hoặc cột nào trong bảng của bạn. Điều này giúp bạn tạo ra một bảng linh hoạt và phù hợp với nhu cầu của bạn.

HBase - Read data

Để đọc dữ liệu trong HBase, bạn có thể sử dụng lệnh `get`. Lệnh `get` cho phép bạn lấy giá trị của một cột cụ thể trong hàng của bảng của bạn. Để sử dụng lệnh `get`, bạn cần chỉ định tên bảng, tên hàng và tên cột.

Cú pháp của lệnh `get` như sau:

```
get 'ten_bang', 'hang_key', 'ten_cot'
```

Trong đó:

- `ten_bang` là tên của bảng bạn muốn lấy dữ liệu.
- `hang_key` là khóa của hàng bạn muốn lấy dữ liệu.
- `ten_cot` là tên của cột bạn muốn lấy dữ liệu.

Ví dụ, để lấy giá trị của cột `age` cho hàng `1` trong bảng `user_table`, bạn có thể sử dụng lệnh sau:

```
get 'user_table', '1', 'info:age'
```

Lệnh này sẽ trả về giá trị của cột `age` cho hàng `1` trong bảng `user_table`.

Nếu bạn muốn lấy nhiều giá trị cho cùng một hàng, bạn có thể sử dụng nhiều lệnh `get` khác nhau. Ví dụ, để lấy giá trị của cột `age` và giá trị của cột `name` cho hàng `1` trong bảng `user_table`, bạn có thể sử dụng lệnh sau:

```
get 'user_table', '1', 'info:age'
get 'user_table', '1', 'info:name'
```

Lệnh này sẽ trả về giá trị của cột `age` và giá trị của cột `name` cho hàng `1` trong bảng `user_table`.

Với lệnh `get`, bạn có thể lấy bất kỳ giá trị nào cho bất kỳ hàng hoặc cột nào trong bảng của bạn. Điều này giúp bạn tạo ra một bảng linh hoạt và phù hợp với nhu cầu của bạn.

HBase - Delete data

Để xóa dữ liệu trong HBase, bạn có thể sử dụng lệnh `delete`. Lệnh `delete` cho phép bạn xóa giá trị của một cột cụ thể trong hàng của bảng của bạn. Để sử dụng lệnh `delete`, bạn cần chỉ định tên bảng, tên hàng và tên cột.

Cú pháp của lệnh `delete` như sau:

```
delete 'ten_bang', 'hang_key', 'ten_cot'
```

Trong đó:

- `ten_bang` là tên của bảng bạn muốn xóa dữ liệu.
- `hang_key` là khóa của hàng bạn muốn xóa dữ liệu.
- `ten_cot` là tên của cột bạn muốn xóa dữ liệu.

Ví dụ, để xóa giá trị của cột `age` cho hàng `1` trong bảng `user_table`, bạn có thể sử dụng lệnh sau:

```
delete 'user_table', '1', 'info:age'
```

Lệnh này sẽ xóa giá trị của cột `age` cho hàng `1` trong bảng `user_table`.

Nếu bạn muốn xóa nhiều giá trị cho cùng một hàng, bạn có thể sử dụng nhiều lệnh `delete` khác nhau. Ví dụ, để xóa giá trị của cột `age` và giá trị của cột `name` cho hàng `1` trong bảng `user_table`, bạn có thể sử dụng lệnh sau:

```
delete 'user_table', '1', 'info:age' delete 'user_table', '1', 'info:name'
```

Lệnh này sẽ xóa giá trị của cột `age` và giá trị của cột `name` cho hàng `1` trong bảng `user_table`.

Với lệnh `delete`, bạn có thể xóa bất kỳ giá trị nào cho bất kỳ hàng hoặc cột nào trong bảng của bạn. Điều này giúp bạn tạo ra một bảng linh hoạt và phù hợp với nhu cầu của bạn.

HBase - Scan table

Để quét dữ liệu trong HBase, bạn có thể sử dụng lệnh `scan`.

Lệnh `scan` cho phép bạn lấy danh sách các hàng và cột trong bảng của bạn. Để sử dụng lệnh `scan`, bạn chỉ cần nhập lệnh sau trong HBase shell:

```
scan 'ten_bang'
```

Trong đó, `ten_bang` là tên của bảng bạn muốn quét. Lệnh này sẽ liệt kê tất cả các hàng và cột trong bảng của bạn.

Bạn cũng có thể sử dụng các tùy chọn bổ sung để hạn chế kết quả của lệnh `scan`. Ví dụ, để chỉ lấy các hàng từ hàng `1` đến hàng `10` trong bảng `user_table`, bạn có thể sử dụng lệnh sau:

```
scan 'user_table', {STARTROW => '1', ENDROW => '10'}
```

Lệnh này sẽ chỉ liệt kê các hàng từ hàng `1` đến hàng `10` trong bảng `user_table`.

Bạn cũng có thể sử dụng các tùy chọn khác để hạn chế kết quả của lệnh `scan`, bao gồm các tùy chọn cho số cột trả về, số hàng trả về, v.v.

Với lệnh `scan`, bạn có thể lấy danh sách các hàng và cột trong bảng của bạn để có cái nhìn tổng quan về dữ liệu của bạn. Điều này giúp bạn đưa ra các quyết định về cách tổ chức và truy xuất dữ liệu của mình trong HBase.

HBase - Count table

Để đếm số hàng trong bảng HBase, bạn có thể sử dụng lệnh `count`.

Lệnh `count` sẽ trả về số hàng trong bảng của bạn. Để sử dụng lệnh `count`, bạn cần chỉ định tên bảng của bạn như sau:

```
count 'ten_bang'
```

Trong đó, `ten_bang` là tên của bảng bạn muốn đếm số hàng.

Ví dụ, để đếm số hàng trong bảng `user_table`, bạn có thể sử dụng lệnh sau:

```
count 'user_table'
```

Lệnh này sẽ trả về số hàng trong bảng `user_table`.

Ngoài ra, bạn có thể sử dụng các tùy chọn bổ sung để hạn chế kết quả của lệnh `count`. Ví dụ, để chỉ đếm số hàng từ hàng `1` đến hàng `10` trong bảng `user_table`, bạn có thể sử dụng lệnh sau:

```
count 'user_table', {STARTROW => '1', ENDROW => '10'}
```

Lệnh này sẽ chỉ đếm số hàng từ hàng `1` đến hàng `10` trong bảng `user_table`.

Với lệnh `count`, bạn có thể đếm số hàng trong bảng của bạn để có cái nhìn tổng quan về dữ liệu của bạn. Điều này giúp bạn đưa ra các quyết định về cách tổ chức và truy xuất dữ liệu của mình trong HBase.

HBase - Truncate table

Trong HBase, khi bạn muốn xóa toàn bộ dữ liệu trong một bảng, không có lệnh `truncate` để làm điều này. Tuy nhiên, bạn có thể sử dụng lệnh `disable` để tạm ngừng bảng và sau đó sử dụng lệnh `deleteall` để xóa toàn bộ dữ liệu trong bảng đó. Sau khi đã xóa toàn bộ dữ liệu, bạn có thể kích hoạt lại bảng.

Để tạm ngừng bảng, bạn có thể sử dụng lệnh `disable` như sau:

```
disable 'ten_bang'
```

Ở đây, `ten_bang` là tên của bảng bạn muốn tạm ngừng.

Sau khi tạm ngừng bảng, bạn có thể sử dụng lệnh `deleteall` để xóa toàn bộ dữ liệu trong bảng. Lệnh `deleteall` sẽ xóa tất cả các cột và hàng trong bảng của bạn. Ví dụ:

```
deleteall 'ten_bang'
```

Lệnh này sẽ xóa toàn bộ dữ liệu trong bảng `ten_bang`.

Cuối cùng, để kích hoạt lại bảng, bạn có thể sử dụng lệnh `enable` như sau:

```
enable 'ten_bang'
```

Lệnh này sẽ kích hoạt lại bảng `ten_bang`.

Lưu ý rằng khi sử dụng lệnh `deleteall`, toàn bộ dữ liệu trong bảng sẽ bị xóa mà không thể khôi phục lại. Do đó, bạn cần chắc chắn rằng bạn đã sao lưu toàn bộ dữ liệu trước khi sử dụng lệnh này.

HBase - Security Grant, Revoke, User_Permission

Trong HBase, bạn có thể sử dụng tính năng phân quyền để kiểm soát quyền truy cập vào dữ liệu. Tính năng này cho phép bạn xác định các quyền truy cập cho người dùng hoặc nhóm người dùng và giới hạn truy cập của họ vào bảng hoặc nhóm cột cụ thể.

Để sử dụng tính năng phân quyền trong HBase, bạn cần kích hoạt tính năng này trên máy chủ HBase của mình. Sau đó, bạn có thể sử dụng lệnh `grant` để cấp quyền truy cập cho người dùng hoặc nhóm người dùng. Ví dụ, để cấp quyền truy cập cho người dùng có tên là `user1` vào bảng `user_table`, bạn có thể sử dụng lệnh sau:

```
grant 'user1', 'RWC', 'user_table'
```

Trong đó, `RWC` là quyền truy cập được cấp cho người dùng. Bạn có thể sử dụng các ký tự sau để xác định quyền truy cập của người dùng:

- `R`: Đọc dữ liệu
- `W`: Ghi dữ liệu
- `C`: Tạo, xóa hoặc sửa chữa cột gia đình
- `A`: Thêm hoặc xóa cột gia đình

Bạn cũng có thể sử dụng các ký tự sau để xác định quyền truy cập của người dùng cho tất cả các bảng:

- `@`: Quyền truy cập toàn bộ hệ thống

Ngoài ra, bạn cũng có thể sử dụng lệnh `revoke` để thu hồi quyền truy cập của người dùng hoặc nhóm người dùng. Ví dụ, để thu hồi quyền truy cập của người dùng có tên là `user1` vào bảng `user_table`, bạn có thể sử dụng lệnh sau:

```
revoke 'user1', 'user_table'
```

Để xem danh sách quyền truy cập hiện có cho người dùng hoặc nhóm người dùng, bạn có thể sử dụng lệnh `user_permission`. Ví dụ, để xem danh sách quyền truy cập hiện có cho người dùng có tên là `user1`, bạn có thể sử dụng lệnh sau:

```
user_permission 'user1'
```

Tính năng phân quyền trong HBase giúp bạn kiểm soát quyền truy cập vào dữ liệu của mình và đảm bảo rằng chỉ những người dùng được cấp quyền mới có thể truy cập vào dữ liệu.