**TRƯỜNG ĐẠI HỌC TÀI NGUYÊN VÀ MÔI TRƯỜNG THÀNH PHỐ HỒ CHÍ MINH**

**KHOA HỆ THỐNG THÔNG TIN VÀ VIỄN THÁM**



**TIỂU LUẬN**

**HỌC PHẦN: HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU**

**ĐỀ TÀI: Survey Of Big Data Storage Technology**

**Nhóm 11**

**Sinh viên thực hiện:**  **Phạm Quang Huy**

**Văn Ngọc Công Sang**

**Âu Dương Đức Trung**

**Lớp** **: 09\_ĐH\_CNTT3**

**Giảng viên hướng dẫn:** **ThS. Phạm Trọng Huynh**

**TP. Hồ Chí Minh, 14 tháng 12 năm 2022**

**MỤC LỤC**

[**1.** Giới thiệu 3](#_Toc121864074)

[**2.** Công nghệ lưu trữ dữ liệu lớn 4](#_Toc121864075)

[2.1. Hệ thống tệp phân tán: 4](#_Toc121864076)

[2.2. Cơ sở dữ liệu NOSQL 8](#_Toc121864077)

[2.3. Cơ sở dữ liệu tất cả trong một Cỗ máy 11](#_Toc121864078)

[2.4. Cụm cơ sở dữ liệu mới của MPP Ngành kiến trúc 12](#_Toc121864079)

[2.5 Tìm hiểu về Apache Spark 13](#_Toc121864080)

[2.6 Những điểm nổi bật của Spark 15](#_Toc121864081)

[**3.** Triển vọng của dữ liệu lớn Công nghệ lưu trữ 16](#_Toc121864082)

[**4.** Demo 17](#_Toc121864083)

# Giới thiệu

Trong vài thập kỷ qua, với việc mở rộng quy mô 1 ứng dụng, các dịch vụ web đã phát triển từ dạng đơn lẻ sang dạng đa phương tiện, dẫn đến cấu trúc và biểu mẫu dữ liệu đa dạng, đồng thời tăng trưởng dữ liệu theo cấp số nhân. International Data Corporation (IDC) dự đoán rằng trong tương lai, kích thước dữ liệu sẽ tăng gấp đôi sau mỗi hai năm [1]. Người tiên phong trong nghiên cứu dữ liệu lớn—McKinsey & Company, một công ty tư vấn ở Hoa Kỳ, định nghĩa dữ liệu lớn là: tập hợp dữ liệu có quy mô vượt quá khả năng thu nhận, lưu trữ, quản lý và phân tích của các công cụ cơ sở dữ liệu thông thường [2]. Hệ thống lưu trữ dữ liệu truyền thống đã đạt đến điểm tắc nghẽn và không thể hoàn thành xử lý dữ liệu kịp thời. Dữ liệu lớn có các đặc điểm như dung lượng lớn, nhiều loại dữ liệu, mật độ giá trị thấp, tốc độ xử lý cao, quan hệ động phức tạp giữa các dữ liệu. Và yêu cầu về tính sẵn sàng cao, khả năng mở rộng và độ tin cậy [3], đặt ra những thách thức đối với công nghệ lưu trữ dữ liệu truyền thống. Nghiên cứu và tổng kết các loại công nghệ lưu trữ dữ liệu mới, cho các bài toán về quy mô ứng dụng

Dự án quỹ: Quỹ Khoa học Tự nhiên Quốc gia Trung Quốc (61462070); Dự án quy hoạch khoa học và công nghệ vùng (20130364). mở rộng, tăng trưởng dữ liệu nhanh chóng, nhiều loại dữ liệu. Đầu tiên, điều tra và phân tích nghiên cứu thực trạng hệ thống tệp phân tán. Bên cạnh đó, phân tích và so sánh đặc điểm của các hệ thống tệp phân tán chính như GFS [4], HDFS [5], GlusterFS [6], GridFS [7], TFS [8], Lustre [9], FastDFS [10]. Ngoài ra, bằng cách nghiên cứu bốn mô hình dữ liệu của cơ sở dữ liệu NoSql, tương ứng với công nghệ lưu trữ của mô hình giá trị khóa, mô hình kiểu cột, mô hình tài liệu và mô hình đồ họa. Nghiên cứu khả năng ứng dụng và tính năng kiến ​​trúc của máy tất cả trong một dữ liệu lớn hiện tại. Cuối cùng, giới thiệu định nghĩa và kịch bản của các loại cụm cơ sở dữ liệu mới sử dụng khung MPP (Xử lý song song lớn) [11] và dự báo các xu hướng nghiên cứu có thể xảy ra của công nghệ lưu trữ trong tương lai.

# Công nghệ lưu trữ dữ liệu lớn

Nghiên cứu và phân tích công nghệ lưu trữ dữ liệu lớn từ bốn khía cạnh sau: hệ thống tệp phân tán, cơ sở dữ liệu NoSQL, công nghệ lưu trữ dữ liệu kiểu mới của kiến ​​trúc MPP và cơ sở dữ liệu máy tất cả trong một. Ngoài ra, còn một số khuyến nghị áp dụng cho các môi trường khác nhau nhằm nắm bắt các trạng thái phát triển của công nghệ lưu trữ dữ liệu từ các góc độ khác nhau.

### 2.1. Hệ thống tệp phân tán:

Hệ thống tập tin là phần cơ bản của chương trình ứng dụng. Tuy nhiên, với việc phát triển ứng dụng trên mạng, dữ liệu phát triển nhanh. Do đó, công nghệ lưu trữ dữ liệu lớn đã trở thành nhiệm vụ chính của các doanh nghiệp và các tổ chức nghiên cứu. Vì dung lượng lưu trữ hạn chế nên các hệ thống lưu trữ truyền thống có thể hầu như không giải quyết được vấn đề lưu trữ dữ liệu lớn. Vì vậy, chúng tôi sử dụng hệ thống tập tin phân tán để chuyển tải hệ thống sang nhiều nút. Các hệ thống tệp phân tán cung cấp dung lượng lưu trữ và băng thông I/O sao cho quy mô của hệ thống có thể dễ dàng mở rộng [12].

Nói chung, việc xác định xem một hệ thống tệp phân tán có thành công hay không phụ thuộc vào ba yếu tố sau: chế độ lưu trữ dữ liệu, tốc độ đọc, cơ chế bảo mật. Tuy nhiên, vẫn còn những cải tiến được thực hiện trong hệ thống tệp phân tán. Ví dụ: GFS và HDFS được thiết kế cho các tệp lớn không thể đáp ứng yêu cầu lưu trữ của nhiều tệp nhỏ. Tần suất truy cập của tệp nhỏ cao, dẫn đến tần suất truy cập vào đĩa cứng cao. Vì vậy, hiệu suất của I/O đã bị giảm xuống [13]. Các tệp nhỏ cũng có thể dẫn đến việc tạo ra một số lượng lớn siêu dữ liệu, điều này sẽ ảnh hưởng đến khả năng khôi phục và quản lý máy chủ siêu dữ liệu, sau đó dẫn đến giảm hiệu suất tổng thể. Hơn nữa, vì tệp tương đối nhỏ nên dễ tạo ra sự phân mảnh tệp, gây lãng phí dung lượng ổ đĩa. Tạo liên kết cho từng tệp sẽ gây ra độ trễ mạng [14].

Bảng 1 tóm tắt và so sánh rộng rãi một số hệ thống tệp phân tán phổ biến, chẳng hạn như GFS, HDFS, TFS, Lustre, v.v.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tên | Phân đoạn tệp | Sao lưu hệ thống | Ưu điểm | Nhược điểm | Trường hợp ứng dụng |
| GFS | Các tệp được lưu trữ trong GFS được chia thành các khối có kích thước cố định | Mỗi khối được sao chép vào nhiều máy chủ khối, lưu 3 bản sao theo mặc định. | GFS sẽ không coi lỗi phần cứng là bất thường. Thông thường, một bản cập nhật được xử lý bằng cách thêm dữ liệu mới thay vì thay đổi dữ liệu hiện có. | Không áp dụng cho lưu trữ tệp nhỏ. Các tệp nhỏ bổ sung sẽ làm giảm hiệu suất. | Tập dữ liệu lớn phân tán lớn. Kích thước dữ liệu thường nằm trong khoảng 4G~40G. |
| HDFS | Các tệp lớn được chia thành một số khối có kích thước mặc định là 64 MB. Mỗi khối sẽ lưu trữ một số bản sao trên nhiều hơn một nút dữ liệu. | Hỗ trợ sao chép dữ liệu. Lưu trữ nhiều bản sao trên các nút khác nhau. | Khả năng mở rộng là rất mạnh mẽ. Một phiên bản HDFS duy nhất có thể hỗ trợ hàng chục triệu tài liệu. Và có khả năng thời gian thực cao. | Không thể sử dụng cho các tình huống yêu cầu độ trễ thấp trong truy cập dữ liệu. Không thể lưu trữ các tệp nhỏ quy mô lớn. | Các tập dữ liệu rất lớn có kích thước từ GB đến TB. |
| GlusterFS | Không hỗ trợ phân chia tập tin | Hỗ trợ sao chép dữ liệu và cung cấp không gian tên toàn cầu. Nhiều bản sao của nhiều tệp có thể được lưu trữ trong các máy chủ khác nhau. Trong khi đọc một bản sao, hệ thống sẽ chọn bản sao gần nhất một cách thuận lợi. | Hỗ trợ CIFS, NFS và máy gốc sử dụng ứng dụng khách GlusterFS. Hệ thống nhiều tệp có thể được triển khai trên hệ thống tệp phân tán ảo. Sử dụng bảng điều khiển dành cho quản trị viên, quản lý cập nhật trung tâm một cách dễ dàng. Tất cả các nút có thể được sử dụng để lấy dữ liệu. | Chỉ có thể quản lý hệ thống trên một máy chủ, không dư thừa. Không thể thêm một nút mới nửa chừng. Không biết cách thêm nhiều đĩa vào mỗi nút; Không có chính sách bảo mật nào trong GlusterFS. | Hỗ trợ lưu trữ tệp lớn ở cấp độ PB. |
| TFS | Không hỗ trợ phân đoạn tập tin. Một số lượng lớn các tệp nhỏ sẽ được hợp nhất thành một tệp lớn. | TFS lưu trữ các tệp dữ liệu theo khối và lưu trữ nhiều bản sao trong trường hợp bảo mật dữ liệu. | Hoạt động đơn giản, với khả năng mở rộng và cân bằng tải mượt mà. Hỗ trợ mở rộng quy mô tuyến tính. Có thể dễ dàng mở rộng đến mức PB. | Khi tính đồng bộ cao và kích thước tệp lớn hơn 5 MB, các lỗi nghiêm trọng sẽ phát sinh trong TFS. Trong những trường hợp hiếm hoi, hỗ trợ lưu trữ tệp lớn. Không hỗ trợ danh mục và sự cho phép của người dùng. | Việc lưu trữ và xử lý một lượng lớn dữ liệu phi cấu trúc và hình ảnh khổng lồ trên trang web taobao. |
| GridFS | Hỗ trợ chia một file lớn thành nhiều file tài liệu nhỏ. | GridFS lưu trữ dữ liệu tệp và siêu dữ liệu tệp trong MongoDB. Sao chép tệp để đối phó với chuyển đổi dự phòng và tích hợp dữ liệu. Và cũng có thể được sử dụng để đọc tiện ích mở rộng, sao lưu nóng hoặc được sử dụng làm nguồn dữ liệu xử lý hàng loạt ngoại tuyến. | Nó dựa trên mô hình cấu trúc của bộ lưu trữ đối tượng, giảm thêm chi phí truy cập trong thời gian chạy. GridFS được thiết kế để điều chỉnh cơ chế truy cập, nhằm thích ứng với hiệu suất ứng dụng nhanh hơn do chế độ I/O đưa ra. Có thể cung cấp hiệu suất I/O nhanh nhất mà cụm ứng dụng cần. GridFS đảm bảo cân bằng tải trong từng thiết bị lưu trữ, thay vì lấp đầy một nút. | Tốc độ đọc tệp từ GridFS chậm hơn so với đọc trực tiếp từ hệ thống tệp. Nếu tệp lớn và được lưu trữ dưới dạng nhiều tệp, không thể khóa tất cả các khối tệp khi sửa đổi tệp lớn này. Khi thay đổi tài liệu được lưu trữ trong GridFS, nó chỉ có thể xóa tài liệu cũ trước rồi lưu lại tài liệu. | Thích hợp cho các tệp lớn hiếm khi cần thay đổi. |
| Lustre | Chia dữ liệu thành một số đối tượng cố định.Mỗi đối tượng chứa một số khối dữ liệu. Khi một khối dữ liệu được ghi vào đối tượng vượt quá dung lượng của nó, lần ghi tiếp theo sẽ được lưu trữ trong đối tượng tiếp theo. Lustre có thể phân phối tệp thành tối đa 160 đối tượng để lưu trữ. Áp dụng mẫu RAIDO trên nhiều đĩa cứng. | Lustre cung cấp hai công cụ sao lưu. Một cái được sử dụng để quét hệ thống tệp và một cái khác được sử dụng để đóng gói sao lưu và thu hồi áp lực. | Có thể cung cấp khả năng chia sẻ dữ liệu và xử lí song song. Khả năng mở rộng rất mạnh. Có thể cung cấp công nghệ chuyển đổi dự phòng cho siêu dữ liệu và dữ liệu mục tiêu dưới sự quản lí của lustre, đạt được quyền truy cập với độ tin cậy cao. Cơ chế quản lý phân tán có thể đạt được kiểm soát đồng thời. Cung cấp quyền truy cập trong nhiều giao thức mạng | Rất khó để thực hiện phản chiếu dữ liệu. Chuyển đổi dự phòng giữa các nút dựa trên công nghệ nhịp tim của bên thứ ba. Chỉ có hai nút quản lý siêu dữ liệu. Nếu quy mô hệ thống đã đạt đến quy mô nhất định, nút quản lý sẽ đạt đến mức quá tải. Lustre kernel chỉ có thể được triển khai trên Linux với một số hạn chế. | Hỗ trợ lưu trữ tệp ở mức PB và phù hợp với cụm máy tính lớn hoặc siêu máy tính |
| Ceph | Phân tán dữ liệu liên tục trên nhiều ổ đĩa để truy cập, nhằm thích nghi với việc cân bằng tải. | Hỗ trợ sao chép dữ liệu. Có nhiều máy chủ siêu dữ liệu. | Lưu trữ dữ liệu và siêu dữ liệu riêng biệt. Quản lý siêu dữ liệu bằng phân phối động. Có bộ nhớ nhờ đối tượng phân tán tự động đáng tin cậy. | Công nghệ vẫn chưa hoàn thiện, không thể được áp dụng vào môi trường sản xuất | Hỗ trợ lưu trữ tệp lớn ở mức PB |
| FastDFS | FastDFS không lưu trữ các tệp trong các khối. Các tệp do máy tính tải lên tương ứng với các tệp được lưu trữ trên máy chủ. | FastDFS áp dụng chế độ lưu trữ của nhóm. Các máy chủ lưu trữ trong cùng một nhóm sao lưu lẫn nhau. | Máy chủ FastDFS chỉ có hai ký tự, trình theo dõi và nút lưu trữ. Vì vậy, nó có tính năng nhẹ. FastDFS áp dụng chế độ lưu trữ nhóm, linh hoạt và được kiểm soát mạnh mẽ.  Trong FastDFS, mỗi nút là nút chính, với cấu trúc ngang hàng. Có thể thay đổi số lượng trình theo dõi bất cứ lúc nào, theo áp lực của máy chủ. | FastDFS không lưu trữ các tệp trong các khối. Vì vậy, nó không phù hợp với trường hợp điện toán phân tán. Dung lượng lưu trữ bị giới hạn bởi một máy chủ lưu trữ duy nhất. | Phù hợp với dịch vụ có lưu lượng truy cập cao, với tệp là nhà cung cấp dịch vụ, chẳng hạn như trang web album ảnh, trang web video, v.v.Và một lượng lớn các tệp nhỏ, 4K~500M. |

### 2.2. Cơ sở dữ liệu NOSQL

Với sự tăng trưởng nhanh chóng về kích thước dữ liệu của người dùng doanh nghiệp và nâng cao nhu cầu của người dùng về mức độ dịch vụ, cơ sở dữ liệu quan hệ truyền thống có một số hạn chế. Cơ sở dữ liệu truyền thống sử dụng tệp phẳng dựa trên bản ghi có cấu trúc để lưu trữ tất cả dữ liệu ứng dụng, dẫn đến sự không khớp giữa ứng dụng và cơ sở dữ liệu. Điều này xảy ra khi ứng dụng được mã hóa bằng ngôn ngữ khai báo. Cấu trúc của nó hoàn toàn khác với các cơ sở dữ liệu này [21]. Trước đây, để cải thiện hiệu suất của hệ thống, các thành phần và tài nguyên được mở rộng theo chiều dọc. Tuy nhiên, do bộ lưu trữ và ứng dụng không còn tách biệt nên mỗi lần mở rộng tài nguyên sẽ làm gián đoạn dịch vụ và đặt lại ứng dụng.

Hầu hết dữ liệu chúng tôi tạo ra là dữ liệu không đồng nhất. Sự tồn tại của một số lượng lớn dữ liệu có cấu trúc và phi cấu trúc gây khó khăn cho việc xác định trước mô hình dữ liệu quan hệ thống nhất và hoàn hảo, đồng thời khả năng mở rộng theo chiều ngang của cơ sở dữ liệu quan hệ là kém [22]. Hầu hết các cơ sở dữ liệu quan hệ không hỗ trợ lưu trữ phân tán quy mô lớn. Đồng thời, khó có thể đáp ứng yêu cầu thời gian thực về tính đồng thời cao và lượng dữ liệu lớn. Vì vậy, công nghệ lưu trữ cơ bản không chỉ linh hoạt để cho phép dữ liệu được lưu trữ ở dạng tự nhiên mà còn đáp ứng nhu cầu của biên giới.

So với cơ sở dữ liệu quan hệ, hệ thống lưu trữ cơ sở dữ liệu NoSQL hỗ trợ lưu trữ và quản lý động dữ liệu khối. Nó tránh được sự phức tạp không cần thiết, có thông lượng cao và có thể xử lý tốt tỷ lệ ngang. Và khả năng chịu lỗi cao có thể lưu trữ dữ liệu có cấu trúc, bán cấu trúc và phi cấu trúc để tránh ánh xạ quan hệ đối tượng. Ý tưởng thiết kế của cơ sở dữ liệu NoSQL là trích xuất cơ chế lập chỉ mục của cơ sở dữ liệu quan hệ, kết hợp chiến lược lưu trữ phân tán và xóa những thứ không cần thiết đối với một số vấn đề trong hệ thống SQL. Do đó, nó đạt được hiệu quả tương đối tốt, khả năng mở rộng và tính linh hoạt [23].

Cơ sở dữ liệu NOSQL chủ yếu được chia thành: lưu trữ khóa-giá trị, lưu trữ dựa trên cột, lưu trữ tài liệu, lưu trữ đồ họa.

Cơ sở dữ liệu khóa-giá trị được thiết kế để hỗ trợ các hoạt động truy vấn đơn giản, để lại các hoạt động phức tạp cho lớp ứng dụng. Tập dữ liệu sẽ ánh xạ khóa tới một hoặc một tập hợp các giá trị. Tức là, chìa khóa là từ khoá duy nhất để tìm mỗi địa chỉ dữ liệu, điều đó cũng có nghĩa là nó rất cần thiết. Giá trị là nội dung mà dữ liệu thực sự lưu trữ. Lưu trữ khóa-giá trị cung cấp một bảng băm với các cặp khóa-giá trị trên các máy chủ từ xa của một cụm phân tán, để triển khai ánh xạ từ khóa đến giá trị. Giá trị băm dựa trên khóa định vị trực tiếp địa chỉ của dữ liệu, đạt được truy vấn đồng thời cao nhanh chóng và cũng hỗ trợ hoạt động của dữ liệu hàng loạt. Lưu trữ khóa-giá trị được chia thành loại khóa-giá trị, loại tài liệu khóa và loại cột khóa (24). Loại khóa cột là sự mở rộng điển hình của các cặp khóa-giá trị của loại khóa-giá trị. Do tính đơn giản và khả năng mở rộng linh hoạt, nó cũng là xu hướng chính của mô hình dữ liệu.

Lưu trữ dựa trên cột tổng quát thay thế các cột bằng họ cột. Ý tưởng của cơ sở dữ liệu quan hệ là lưu trữ tất cả các bảng bằng một dòng trên đĩa. Nghĩa là, một danh sách các mục được liên kết với cùng một id hàng cụ thể sẽ được lưu trữ cùng nhau [25]. Vì các ngân hàng hoặc tổ chức tài chính cần duy trì một số lượng lớn các bản ghi liên quan, không đảm bảo rằng tất cả các giá trị luôn được lưu trữ một cách liên tục. Trong cơ sở dữ liệu của dữ liệu cột, toàn bộ cột của bảng được lưu trữ cùng nhau, được ánh xạ tới một khóa. Vì tất cả các mục được liệt kê đều có chỉ mục nên chúng tôi chỉ có thể tìm kiếm một phần của bảng. Một cột cũng có thể có các cột lồng nhau theo cấu trúc phân cấp và một trong số đó là siêu cột [26]. Điều này cung cấp truy vấn đơn giản và truy cập nhanh, đồng thời tránh được chi phí không cần thiết khi tìm khóa duy nhất của bản ghi.

Cơ sở dữ liệu tài liệu là một loại cơ sở dữ liệu quan hệ khác, được sử dụng để lưu trữ dữ liệu bán cấu trúc. Có XML (ngôn ngữ đánh dấu có thể mở rộng), JSON (ký hiệu đối tượng JavaScript), hoặc các định dạng tương tự khác [27]. Một tài liệu có thể được xem như một dòng trong cơ sở dữ liệu quan hệ, chứa tất cả các thông tin liên quan của tài liệu. Một nhóm bao gồm nhiều tài liệu và mỗi tài liệu có thể có các mẫu khác nhau, số lượng và kiểu lưu trữ dữ liệu khác nhau [28]. Lưu trữ tin nhắn văn bản là tối ưu hóa đặc biệt. Do các tập dữ liệu liên quan được lưu trữ chuyên sâu, chi phí của hoạt động SQL JOIN sẽ được tiết kiệm. Mặc dù cơ sở dữ liệu là thiết kế không có giản đồ, nhưng nó lưu trữ các bản ghi bán cấu trúc và có phân cấp kết cấu.

Cơ sở dữ liệu đồ thị phù hợp nhất với tìm kiếm ứng dụng và truyền tải, chẳng hạn như tìm các liên kết có liên quan trên LinkedIn, tìm bạn bè trên Facebook [29], v.v. Nó chú ý nhiều hơn đến mối quan hệ giữa các mục dữ liệu hơn là bản thân dữ liệu. Họ tối ưu hóa khả năng truyền tải nhanh chóng và sử dụng thuật toán đồ thị một cách hiệu quả. Ví dụ: con đường ngắn nhất trước tiên là để tìm sự liên quan giữa các thông tin, v.v.

Bảng 2 phân tích và kết luận các kiểu lưu trữ chính của cơ sở dữ liệu NoSQL:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Kiểu cơ sở dữ liệu | Ưu điểm | Nhược điểm | Mô hình dữ liệu | Trường hợp ứng dụng | Ứng dụng |
| Lưu trữ khóa-giá trị | Có hiệu suất đọc và ghi đồng thời rất cao. Dữ liệu được lập chỉ mục và phân đoạn theo giá trị chính. Tìm kiếm nhanh chóng và mô hình dữ liệu đơn giản. | Dữ liệu không có cấu trúc và không hỗ trợ hoạt động dữ liệu logic phức tạp. | ánh xạ khóa-giá trị giữa khóa và giá trị | bộ đệm nội dung. Chủ yếu được sử dụng cho hệ thống nhật ký. | Dynamo,  Redis,  Voldemort |
| Lưu trữ dựa trên cột | Tìm kiếm nhanh, khả năng mở rộng tốt và tiết kiệm nhiều thao tác I/O. Nó dễ dàng hơn cho phần mở rộng phân tán. | Chức năng tương đối hạn chế. | Dựa trên cột lưu trữ, nơi dữ liệu trong cùng một cột được lưu trữ trên cùng một trang | hệ thống tập tin phân tán | Bigtable,  Cassandra,  HBase,  HyperTable |
| Lưu trữ tài liệu | Không cần xác định trước cấu trúc dữ liệu. Sử dụng tài liệu có định dạng cụ thể thay vì bộ làm đơn vị lưu trữ dữ liệu. | Hiệu quả truy vấn chưa cao, thiếu cú ​​pháp truy vấn thống nhất. | Giá trị trỏ đến dữ liêu có cấu trúc | Ứng dụng Web | CouchDB,  MongoDB  XML Database  ThruDB |
| Lưu trữ đồ họa | Sử dụng lý thuyết đồ thị và thuật toán liên quan để cải thiện hiệu suất lưu trữ, dữ liệu quản lý và vận hành. | Chức năng tương đối hạn chế., | Cấu trúc đồ thị | Mạng xã hội, biểu đồ mối quan hệ | Neo4j, GraphDB InfoGrdi |

### Cơ sở dữ liệu tất cả trong một Cỗ máy

Trong những năm gần đây, đối mặt với việc xử lý và lưu trữ dữ liệu hàng loạt, nhiều nhà sản xuất phần cứng truyền thống đề xuất giải pháp tích hợp --- máy tất cả trong một cơ sở dữ liệu, đã trở thành một điểm nóng. Bằng hình thức sản phẩm của máy tất cả trong một, nó đơn giản hóa sự phức tạp của việc triển khai và quản lý cơ sở hạ tầng của dữ liệu trung tâm, giải quyết các vấn đề của liên tục mở rộng tài nguyên phần cứng cơ bản trong thời đại dữ liệu lớn, yêu cầu của máy tất cả trong một và chi phí lưu trữ dữ liệu lớn. Các nhà sản xuất quốc tế, chẳng hạn như IBM, Oracle, EMC, tung ra các sản phẩm và giải pháp tích hợp cho dữ liệu lớn [35]. Theo sau họ, các nhà sản xuất Trung Quốc cũng phát triển cơ sở dữ liệu máy tính tất cả trong một của riêng mình. Ví dụ: máy tất cả trong một cơ sở dữ liệu của Huawei sử dụng lợi thế kiến trúc phần cứng của nó là tính toán, lưu trữ và hội tụ mạng, cũng như tính năng của thông lượng cao và IOPS cao, tích hợp Xuất sắc đặc điểm của thông minh card mạng, SSD và các phần cứng khác, giải quyết tình trạng tắc nghẽn hiệu suất giữa tính toán và lưu trữ. Bộ xử lý dữ liệu lớn XData của Shuguang phân tách đơn vị lưu trữ dữ liệu và đơn vị xử lý. Bằng cách xây dựng phần mềm trung gian dịch vụ hiệu quả, trùng hợp nút lưu trữ dữ liệu cơ bản áp dụng cấu trúc không chia sẻ thành một hình ảnh hệ thống xử lý dữ liệu duy nhất. Máy đa năng đám mây dữ liệu lớn Langchao bao gồm các phiên kỹ thuật như lưu trữ dữ liệu, xử lý dữ liệu, trình bày dữ liệu, v.v. Ngoài ra còn có máy đa năng điện toán đám mây khối dữ liệu Yunchuang Storage, máy tất cả trong một dữ liệu lớn Zhongzhiheda, máy tất cả trong một dựa trên dữ liệu lớn Zhiyitu Hadoop [36].

Máy tất cả trong một cơ sở dữ liệu thường phù hợp với mô hình dữ liệu có quan hệ lưu trữ phức tạp. Đồng thời, điện toán cần tính giao dịch cao và nhất quán. Nói chung, cấu hình máy chủ công cụ cơ sở dữ liệu phụ thuộc vào nhu cầu đồng thời và cấu hình máy chủ nút lưu trữ cơ sở dữ liệu phụ thuộc vào yêu cầu kích thước dữ liệu [37]. Máy tất cả trong một cơ sở dữ liệu áp dụng kiến trúc xử lý dữ liệu lớn phân tán đầy đủ, tích hợp phần cứng và phần mềm trong một hệ thống. Với sự phát triển của dữ liệu người dùng và mở rộng kinh doanh, nó có thể được cải thiện bằng cách mở rộng chiều dài phần cứng và cũng có thể đạt được tỷ lệ tuyến tính bằng cách thêm các nút theo chiều rộng, đảm bảo hiệu suất của độ trễ thấp, thông lượng cao và tính liên tục của doanh nghiệp [38]. Máy tất cả trong một là sự kết hợp giữa phần mềm và phần cứng, được thiết kế hoàn toàn để xử lý lưu trữ dữ liệu hàng loạt. Và nó được cấu thành từ một bộ máy chủ tích hợp, thiết bị lưu trữ, hệ điều hành, hệ quản trị cơ sở dữ liệu và phần mềm quản lý dữ liệu được cài đặt sẵn. Nó cung cấp giải pháp lưu trữ dữ liệu lớn, chủ yếu cho thị trường kho dữ liệu lớn. Và khả năng thông lượng cao của nó tạo điều kiện giải quyết vấn đề tắc nghẽn I/O.

Người dùng có thể lựa chọn các dòng sản phẩm khác nhau tùy theo yêu cầu, tùy chỉnh theo yêu cầu.

Tuy nhiên, cơ sở dữ liệu tất cả trong một máy cũng phải đối mặt với những thách thức. Trong thời đại dữ liệu lớn, lượng dữ liệu ngày càng tăng lên một cách đáng kinh ngạc. Do đó, nếu người dùng cần mở rộng máy tất cả trong một, họ chỉ có thể thêm một tủ thiết bị, dẫn đến việc mở rộng không linh hoạt. Và bởi vì phần mềm tất cả trong một có tính tích hợp cao, nên nó khó có thể được triển khai trong các môi trường khác.

Trong một số ngành, nhu cầu thay đổi nhanh chóng hơn. Vì vậy, mô hình kinh doanh sẽ thay đổi rất nhanh với nó. Việc sử dụng máy tất cả trong một sẽ hạn chế các thao tác của doanh nghiệp ngược lại. Nhưng trong một số ứng dụng tương đối thuần thục và ổn định, máy tất cả trong một thể hiện giá trị của việc đơn giản hóa Công nghệ thông tin.

### Cụm cơ sở dữ liệu mới của MPP Ngành kiến trúc

MPP là hệ thống xử lý song song quy mô lớn, là một loại phương pháp mở rộng tài nguyên hệ thống, chủ yếu là xử lý song song. Điều này có nghĩa là một máy tính có nhiều bộ xử lý mạng [39].

Mở rộng theo chiều ngang là mục tiêu thiết kế chính của cơ sở dữ liệu kiến trúc MPP. Nó được liên kết bởi nhiều máy chủ SMP thông qua internet của các nút cố định, cộng tác cho các tác vụ phổ biến.

Từ cấp độ người dùng, nó là một hệ thống máy chủ, hỗ trợ nghiêm ngặt dữ liệu mô hình quan hệ. Đặc điểm lớn nhất là mỗi nút chỉ có thể truy cập tài nguyên cục bộ riêng của họ, không có chia sẻ.

Cụm cơ sở dữ liệu sử dụng kiến trúc MPP có thể hỗ trợ hiệu quả cho việc lưu trữ dữ liệu có cấu trúc khối lượng lớn ở cấp độ PB. Nó dựa trên kiến trúc Shared nothing. Bằng công nghệ xử lý dữ liệu lớn của lưu trữ cột, chỉ mục chi tiết thô, v.v. và kết hợp mô hình điện toán phân tán với hiệu suất cao, nó hoàn thành hỗ trợ kỹ thuật cho ứng dụng lưu trữ của lớp phân tích. Môi trường hoạt động chủ yếu là máy tính giá rẻ và nó có ưu điểm là hiệu suất cao và khả năng mở rộng cao [40]. Nó có thể cải thiện hiệu suất xử lý dữ liệu, cải thiện tải quy trình dữ liệu, nâng cao hiệu quả xử lý dữ liệu hàng loạt và giảm tổng chi phí xử lý mỗi TB. Do đó nó đã được sử dụng rộng rãi trong kho dữ liệu doanh nghiệp thế hệ mới và phân tích dữ liệu có cấu trúc đồng ruộng.

### 2.5 Tìm hiểu về Apache Spark

Ngày nay có rất nhiều hệ thống đang sử dụng Hadoop để phân tích và xử lý dữ liệu lớn. Ưu điểm lớn nhất của Hadoop là được dựa trên một mô hình lập trình song song với xử lý dữ liệu lớn là MapReduce, mô hình này cho phép khả năng tính toán có thể mở rộng, linh hoạt, khả năng chịu lỗi, chi phí rẻ. Điều này cho phép tăng tốc thời gian xử lý các dữ liệu lớn nhằm duy trì tốc độ, giảm thời gian chờ đợi khi dữ liệu ngày càng lớn.

Dù có rất nhiều điểm mạnh về khả năng tính toán song song và khả năng chịu lỗi cao nhưng Apache Haddop có một nhược điểm là tất cả các thao tác đều phải thực hiện trên ổ đĩa cứng điều này đã làm giảm tốc độ tính toán đi gấp nhiều lần.

Để khắc phục được nhược điểm này thì Apache Spark được ra đời. Apache Spark có thể chạy nhanh hơn 10 lần so với Haddop ở trên đĩa cứng và 100 lần khi chạy trên bộ nhớ RAM.

#### 2.5.1 Apache Spark là gì ?

Apache Spark là một framework mã nguồn mở tính toán cụm, được phát triển sơ khởi vào năm 2009 bởi AMPLab. Sau này, Spark đã được trao cho Apache Software Foundation vào năm 2013 và được phát triển cho đến nay.

Tốc độ xử lý của Spark có được do việc tính toán được thực hiện cùng lúc trên nhiều máy khác nhau. Đồng thời việc tính toán được thực hiện ở bộ nhớ trong (in-memories) hay thực hiện hoàn toàn trên RAM.

Spark cho phép xử lý dữ liệu theo thời gian thực, vừa nhận dữ liệu từ các nguồn khác nhau đồng thời thực hiện ngay việc xử lý trên dữ liệu vừa nhận được ( Spark Streaming).

Spark không có hệ thống file của riêng mình, nó sử dụng hệ thống file khác như: HDFS, Cassandra, S3,…. Spark hỗ trợ nhiều kiểu định dạng file khác nhau (text, csv, json…) đồng thời nó hoàn toàn không phụ thuộc vào bất cứ một hệ thống file nào.

#### 2.5.2 Thành phần của Spark

**Apache Spark** gồm có 5 thành phần chính : Spark Core, Spark Streaming, Spark SQL, MLlib và GraphX, trong đó:

* **Spark Core** là nền tảng cho các thành phần còn lại và các thành phần này muốn khởi chạy được thì đều phải thông qua Spark Core do Spark Core đảm nhận vai trò thực hiện công việc tính toán và xử lý trong bộ nhớ (In-memory computing) đồng thời nó cũng tham chiếu các dữ liệu được lưu trữ tại các hệ thống lưu trữ bên ngoài.
* **Spark SQL** cung cấp một kiểu data abstraction mới (SchemaRDD) nhằm hỗ trợ cho cả kiểu dữ liệu có cấu trúc (structured data) và dữ liệu nửa cấu trúc (semi-structured data – thường là dữ liệu dữ liệu có cấu trúc nhưng không đồng nhất và cấu trúc của dữ liệu phụ thuộc vào chính nội dung của dữ liệu ấy). **Spark SQL** hỗ trợ DSL (Domain-specific language) để thực hiện các thao tác trên DataFrames bằng ngôn ngữ Scala, Java hoặc Python và nó cũng hỗ trợ cả ngôn ngữ SQL với giao diện command-line và ODBC/JDBC server.
* **Spark Streaming** được sử dụng để thực hiện việc phân tích stream bằng việc coi stream là các mini-batches và thực hiệc kỹ thuật RDD transformation đối với các dữ liệu mini-batches này. Qua đó cho phép các đoạn code được viết cho xử lý batch có thể được tận dụng lại vào trong việc xử lý stream, làm cho việc phát triển lambda architecture được dễ dàng hơn. Tuy nhiên điều này lại tạo ra độ trễ trong xử lý dữ liệu (độ trễ chính bằng mini-batch duration) và do đó nhiều chuyên gia cho rằng Spark Streaming không thực sự là công cụ xử lý streaming giống như Storm hoặc Flink.
* **MLlib** (Machine Learning Library): MLlib là một nền tảng học máy phân tán bên trên Spark do kiến trúc phân tán dựa trên bộ nhớ. Theo các so sánh benchmark Spark MLlib nhanh hơn 9 lần so với phiên bản chạy trên Hadoop (Apache Mahout).
* **GrapX**: Grapx là nền tảng xử lý đồ thị dựa trên Spark. Nó cung cấp các Api để diễn tảcác tính toán trong đồ thị bằng cách sử dụng Pregel Api.

### 2.6 Những điểm nổi bật của Spark

* Xử lý dữ liệu: Spark xử lý dữ liệu theo lô và thời gian thực
* Tính tương thích: Có thể tích hợp với tất cả các nguồn dữ liệu và định dạng tệp được hỗ trợ bởi cụm Hadoop.
* Hỗ trợ ngôn ngữ: hỗ trợ Java, Scala, Python và R.
* Phân tích thời gian thực:
  + Apache Spark có thể xử lý dữ liệu thời gian thực tức là dữ liệu đến từ các luồng sự kiện thời gian thực với tốc độ hàng triệu sự kiện mỗi giây. Ví dụ: Data Twitter chẳng hạn hoặc luợt chia sẻ, đăng bài trên Facebook. Sức mạnh Spark là khả năng xử lý luồng trực tiếp hiệu quả.
  + Apache Spark có thể được sử dụng để xử lý phát hiện gian lận trong khi thực hiện các giao dịch ngân hàng. Đó là bởi vì, tất cả các khoản thanh toán trực tuyến được thực hiện trong thời gian thực và chúng ta cần ngừng giao dịch gian lận trong khi quá trình thanh toán đang diễn ra.
* Mục tiêu sử dụng:
  + Xử lý dữ liệu nhanh và tương tác
  + Xử lý đồ thị
  + Công việc lặp đi lặp lại
  + Xử lý thời gian thực
  + joining Dataset
  + Machine Learning
  + Apache Spark là Framework thực thi dữ liệu dựa trên Hadoop HDFS. Apache Spark không thay thế cho Hadoop nhưng nó là một framework ứng dụng. Apache Spark tuy ra đời sau nhưng được nhiều người biết đến hơn Apache Hadoop vì khả năng xử lý hàng loạt và thời gian thực.

# Triển vọng của dữ liệu lớn Công nghệ lưu trữ

Do số lượng lớn dữ liệu phi cấu trúc và bán cấu trúc, cơ sở dữ liệu quan hệ truyền thống đã bất lực. Tuy nhiên, công nghệ lưu trữ mới, chẳng hạn như Cơ sở dữ liệu NoSQL và hệ thống tệp phân tán vượt trội so với lưu trữ truyền thống, bất kể khả năng chịu lỗi, khả năng mở rộng và tính di động của dữ liệu. Và nó phù hợp để lưu trữ dữ liệu liên tục và quản lý lưu trữ dữ liệu lớn [41]. Nhưng đối với hiệu suất xử lý dữ liệu theo thời gian thực, có một khoảng cách nhất định giữa công nghệ lưu trữ mới và cơ sở dữ liệu quan hệ. Nên cái nào cũng đều có mặt tốt của nó. Hiện nay, sự kết hợp giữa cơ sở dữ liệu quan hệ và hệ thống xử lý song song phân tán có thể cải thiện hiệu quả lưu trữ, tốc độ xử lý và tốc độ phân tích [42]. Cách làm này cũng là xu hướng hot trong tương lai. Vấn đề cốt lõi của công nghệ lưu trữ dữ liệu lớn là hiệu năng. Một kỹ thuật và nền tảng đơn lẻ không còn có thể đáp ứng nhu cầu tăng trưởng dữ liệu bùng nổ và yêu cầu phân tích và lưu trữ dữ liệu từ các nhà khai thác. Trong quá trình phát triển tiếp theo, cơ sở dữ liệu kiểu mới sẽ dần dần được trộn lẫn với hệ sinh thái Hadoop hoặc hệ sinh thái Spark [43], cung cấp SQL và hỗ trợ giao dịch cho ứng dụng. Sử dụng Hadoop hoặc Spark để xử lý dữ liệu bán cấu trúc, phi cấu trúc. Vì vậy trong tương lai, lưu trữ cũng sẽ được phát triển theo hướng kết hợp giữa cụm cơ sở dữ liệu song song MPP và cụm Hadoop/Spark. Ngoài ra, với tốc độ phát triển bùng nổ của dữ liệu doanh nghiệp, máy cơ sở dữ liệu lớn tất cả trong một chắc chắn sẽ trở thành công nghệ hot và được phổ biến rộng rãi.

Bằng cách nghiên cứu công nghệ lưu trữ dữ liệu mới, cho thấy tóm tắt và đối chiếu hệ thống tệp phân tán, cơ sở dữ liệu NoSQL, máy cơ sở dữ liệu tất cả trong một và cụm cơ sở dữ liệu kiểu mới của kiến trúc MPP từ các góc độ khác nhau. Và xu hướng nghiên cứu trong tương lai đã được đưa ra. Lưu trữ dữ liệu lớn vẫn đang trong giai đoạn phát triển nhanh chóng. Không gian phát triển còn rất lớn, cần các nhà nghiên cứu khám phá không ngừng.

# Demo

**#Khởi tạo một Session trong SparkContext để hoạt động được trong môi trường này:**

from pyspark.sql import SparkSession

spark=SparkSession.builder.master("local[\*]").appName('demo').getOrCreate()

**#Tạo khung dữ liệu từ danh sách các hàng**

from datetime import datetime, date

from pyspark.sql import Row

df = spark.createDataFrame([

    Row(MSSV =1122, HOTEN = 'PHAM QUANG HUY', NGAYSINH=date(2002,4,13), DIACHI='QUANG NGAI'),

    Row(MSSV =4532, HOTEN = 'AU DUONNG DUC TRUNG', NGAYSINH=date(2002,2,22), DIACHI ='TP.HO CHI MINH'),

    Row(MSSV =7653, HOTEN = 'VAN NGOC CONG SANG ', NGAYSINH = date(2002,10,5), DIACHI ='LONG AN')

])

df.show()

**#Thêm một bảng vào cluster chúng ta có thể đọc từ TT\_CHUYENBAY.csv()**

data  =  spark.read.csv('/content/TT\_CHUYENBAY.csv', sep= ',', header = True)

data.show(50)

**#hiển thị lược đồ của DataFrame**

data.printSchema()

**#Trả về tất cả tên cột dưới dạng danh sách**

data.columns

**#Một số lệnh biến đổi dữ liệu của spark DataFram**

**#Tạo một bảng tạm thời trên danh mục các chuyến bay khung dữ liệu cục bộ dưới dạng bảng tạm thời mới trên danh mục**

data.createOrReplaceTempView('TT\_CHUYENBAY\_temp')

flights\_10 = spark.sql('SELECT \* FROM TT\_CHUYENBAY\_temp WHERE AIR\_TIME > 10')

flights\_10.show(5)

**#Bổ sung thêm 1 trường mới là HOUR\_ARR(Số h di chuyển= Thời gian cất cánh/60)**

CHUYENBAY = data.withColumn('HOUR\_ARR', data.AIR\_TIME/60)

CHUYENBAY.show()

**#Tạo ra trường VTTB\_BAY tính vận tốc trung bình của các máy bay**

**#(khoảng cách/Số h di chuyển )**

VTTB\_BAY = CHUYENBAY.select(

    "ORIGIN\_AIRPORT", "DESTINATION\_AIRPORT","TAIL\_NUMBER",

    (CHUYENBAY.DISTANCE/CHUYENBAY.HOUR\_ARR).alias("VTTB\_BAY"))

VTTB\_BAY.show(10)

**#Lọc những chuyến bay xuất phát từ sân bay SEA và điểm đến là ANC**

filter\_SEA\_ANC = data.filter("ORIGIN\_AIRPORT == 'SEA'").filter("DESTINATION\_AIRPORT == 'ANC'")

filter\_SEA\_ANC.show()

Tài liệu tham khảo:

[1] Zhang X, Xu F. Survey of Research on Big Data Storage [C] // International Symposium on Distributed Computing and Applications To Business, Engineering & Science. IEEE Computer Society, 2013: 76-80.

[2] Biesdorf S, Court D, Willmott P. Big data: What's your plan? [J]. Mckinsey Quarterly, 2013 (2): 40-51.

[3] TU Xinli, Liu Bo, Lin Weiwei. Survey of Big Data [J]. Application Research of Computers, 2014, 31 (6): 1612-1616.

[4] Garcia H, Ludu A. The Google file system [C] // Acm Sigops Operating Systems Review. ACM, 2003: 29-43.

[5] Tong Ming. Research and application of distributed storage based on HDFS [D]. Huazhong University of Science and Technology, 2012.

[6] Davies A, Orsaria A. Scale out with GlusterFS [J]. Linux Journal, 2013, 2013 (235): 1.

[7] Hows D, Membrey P, Plugge E, et al. GridFS [M]. Apress, 2013.

[8] Zhao Yang. Depth Profiles of TaoBao TFS [J]. Digital Users, 2013 (3).

[9] Wang Bo, Li Xianguo, Zhang Xiao. Research on performance optimization of Lustre file system [J]. Microcomputer Applications, 2011, 27 (5): 31-33.

[10] Yu Qing. Analyses of Distributed File System FastDFS Architecture [J]. Programmer, 2010 (11): 63-65.

[11] Golov N, Rönnbäck L. Big Data Normalization for Massively Parallel Processing Databases [C] // International Workshop on Modeling & Management of Big Data. 2015.

[12] Thereska E, Gunawardena D S, Scott J W, et al. Distributed File System: US, US20120254116 [P]. 2012.

[13] Li Hongqi, Zhu Liping, Sun Guoyu, et al. Design and Implementation of Distributed Storage System Facing Vast Small Files [J]. Computer Engineering and Design, 2016 (1): 86-92.

[14] Qi Ying. Research on Low Latency Access Technology of Distributed File System with Vast Small Files [D]. University of Chinese Academy of Sciences, 2013.

[15] Weil S A, Brandt S A, Miller E L, et al. Ceph: A Scalable, High-Performance Distributed File System [C] // 7th Symposium on Operating Systems Design and Implementation (OSDI '06), November 6-8, Seattle, WA, USA. 2006: 307--320.

[16] Gpfs B. A Shared-Disk File System for Large Computing Clusters [C] // of the First Conference on File and Storage Technologies. 2010.

[17] Xu Chunling, Zhang Guangquan. Comparison and Analysis of Distributed File System Hadoop HDFS and Traditional File System Linux FS [J]. Journal of Soochow University (Engineering Science Edition), 2010, 30 (4): 5-9.

[18] Xiong Wen, Yu Zhibin, Xu Chengzhong. Feature Analysis and Performance Comparison of Several Common Distributed File System [J]. Journal of Integration Technology, 2012, 1 (4): 58-63.

[19] Sawicki A, Nowak T. NETWORK DISTRIBUTED FILE SYSTEM:, US20080320097[P]. 2008.

[20] Shi Xiaodong. Research on High Availability of Distributed File System [D]. Institute of Computing Technology, Chinese Academy of Sciences, 2002.

[21] Qin Xiongpai, Wang Huiju, Du Xiaoyong, et al. big data analytics --Competition and Coexistence of RDBMS and MapReduce [J]. Journal of Software, 2012, 23 (1): 32-45.

[22] Shen Derong, Yu Ge, Wang Xite, et al. Survey of Research on NoSQL System Supporting Big Data Management [J]. Journal of Software, 2013 (8): 1786-1803.

[23] Curé O, Kerdjoudj F, Faye D, et al. On The Potential Integration of an Ontology-Based Data Access Approach in NoSQL Stores [J]. International Journal of Distributed Systems & Technologies, 2012, 4 (3): 166-173.

[24] Wang Jieping, Li Haibo, Song Jie, et al. Research on Cloud Data Storage and Management Standardization [J]. Information Technology and Standardization, 2011 (9): 28-31.

[25] Liu Y, Zhu L, Jiang W. Column caching mechanism for column based database:, EP2743839 [P]. 2014.

[26] Bhogal J, Choksi I. Handling Big Data Using NoSQL [C]// IEEE International Conference on Advanced Information NETWORKING and Applications Workshops. IEEE, 2015: 393-398.

[27] Amirian P, Basiri A, Winstanley A. Efficient Online Sharing of Geospatial Big Data Using NoSQL XML Databases [C] // Fourth International Conference on Computing for Geospatial Research and Application. IEEE, 2013: 152-152.

[28] Deka G C. A Survey of Cloud Database Systems [J]. It Professional, 2014, 16 (2): 50-57.

[29] Castelltort A, Laurent A. Fuzzy Historical Graph Pattern Matching A NoSQL Graph Database Approach for Fraud Ring Resolution [M] // Artificial Intelligence Applications and Innovations. Springer International Publishing, 2015.

[30] Dong-Hai L U, Xian-Bo H E. The Analysis of NoSQL Database [J]. Science & Technology of West China, 2011.

[31] Srivastava P P, Goyal S, Kumar A. Analysis of various NoSql database [C] // International Conference on Green Computing and Internet of Things. IEEE, 2015: 539-544.

[32] Chandra D G. BASE analysis of NoSQL database [J]. Future Generation Computer Systems, 2015, 52: 13–21.

[33] Gu Y, Wang X, Shen S, et al. Analysis of data replication mechanism in NoSQL database MongoDB [C] // IEEE International Conference on Consumer Electronics - Taiwan. IEEE, 2015.

[34] Han J, Haihong E, Le G, et al. Survey on NoSQL database [C] // Pervasive Computing and Applications (ICPCA), 2011 6th International Conference on. IEEE, 2011: 363-366.

[35] Hinshaw F D, Meyers D L, Zane B M. Programmable streaming data processor for database appliance having multiple processing unit groups: US, US7577667[P]. 2009.

[36] Zhang Dong, Qi Kaiyuan, Wu Nan, et al. Architecture and Key Technology of Yunhai Big Data All-in-one Machine [J]. Computer Research and Development, 2016, 53 (2): 374-389.

[37] Yue Junfeng, Zhao Junfeng, Zhao Wei, et al. Analysis of Database All-in-one Machine Technical Architecture [J]. Electric Power Information and Communication Technology, 2013, 11 (4): 60-64.

[38] Pu Siyu, Bai qionghua. A Kind of New-type Cloud Storage All-in-one Machine Backing Up Data Information Automatically:, CN204305087U[P]. 2015.

[39] Li C, Yang J, Han J, et al. The Distributed Storage System Based on MPP for Mass Data [C] // Proceedings of the 2012 IEEE Asia-Pacific Services Computing Conference). IEEE Computer Society, 2012: 384-387.

[40] Chen Z, Song L. A solution based on the MPP'S to storage mass data [C] // Mechatronic Science, Electric Engineering and Computer (MEC), 2011 International Conference on. IEEE, 2011: 868-871.

[41] Cheng Lianjuan. Application Practice and Beneficial Reference of Big Data Promotion in the U. S.: An Analysis from the Perspective of Library [J]. Information & Documentation Services, 2013, 34 (5): 110-112.

[42] Li G. Research Status and Scientific Thinking of Big Data [J]. Bulletin of Chinese Academy of Sciences, 2012.

[43] Chen Jirong, Yue Jiajin. Survey of Big Data Solution Based on Hadoop Ecosystem [J]. Computer Engineering and Science, 2013, 35 (10): 25-35