**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA HỆ THỐNG THÔNG TIN**

**Logo, icon

Description automatically generated**

**BÀI TẬP LỚN 2**

**CƠ CHẾ PHÂN TÁN TRÊN HỆ QUẢN TRỊ NOSQL: COUCHDB**

**MÔN HỌC: CƠ SỞ DỮ LIỆU PHÂN TÁN**

**LỚP IS211.N11. HTCL – NHÓM 15**

**GVHD: NGUYỄN MINH NHỰT**

**THÀNH VIÊN:**

| Đỗ Đặng Kiến Nam | : | 20521627 |
| --- | --- | --- |
| Phạm Thanh Nhựt | : | 20521728 |
| Lê Anh Thư | : | 20521985 |
| Lê Phương Tuyết | : | 20522135 |

**TP. HỒ CHÍ MINH, 2022**

**LỜI CẢM ƠN**

Lời đầu tiên, nhóm chúng em xin đặc biệt gửi lời cảm ơn đến Thầy Nguyễn Minh Nhựt (giảng viên thực hành môn Cơ sở dữ liệu phân tán), người đã dùng tri thức và tâm huyết của mình để truyền đạt cho chúng em vốn kiến thức vô cùng quý báu trong khoảng thời gian học tập. Những kiến thức mà Thầy truyền đạt là bước đệm quan trọng giúp chúng em có thể hoàn thành đề tài đồ án tốt hơn. Xuất phát từ mục đích học tập, tìm hiểu về kiến thức về cơ sở dữ liệu NoSQL, nhóm đã quyết định chọn đề tài “Cơ sở dữ liệu NoSQL CouchDB”. Trong quá trình thực hiện đồ án, dựa trên những kiến thức được Thầy chỉ dạy trên lớp cùng với việc học nhóm và tự tìm hiểu những công cụ và kiến thức mới, nhóm đã cố gắng thực hiện đồ án một cách tốt nhất. Tuy nhiên, trong quá trình thực hiện, không tránh khỏi những sai sót. Hơn nữa, năng lực và thời gian của cá nhân mỗi thành viên trong nhóm là giới hạn nên sản phẩm vẫn chỉ mang tính chất là một đồ án môn học, chưa thực sự hoàn thiện. Do đó, rất mong nhận được những sự góp ý từ Thầy nhằm giúp nhóm hoàn thiện những kiến thức đã học tập và cũng là hành trang để nhóm thực hiện tiếp các đề tài khác trong tương lai. Nhóm rất trân trọng và cảm ơn những kiến thức và kỹ năng được truyền đạt đã giúp mỗi thành viên trong nhóm hoàn thiện vốn kiến thức hiện có để nhóm có thể tiếp tục hoàn thành những đồ án và sản phẩm khác cũng như áp dụng vào công việc trong tương lai.

Một lần nữa xin gửi đến thầy cô, bạn bè lời cảm ơn chân thành và tốt đẹp nhất!

Thành phố Hồ Chí Minh, tháng 12 năm 2021

Nhóm sinh viên thực hiện

MỤC LỤC

DANH MỤC HÌNH

|  |  |
| --- | --- |
|  |  |

DANH MỤC TỪ VIẾT TẮT

|  |  |
| --- | --- |
|  |  |
|  |  |

**CHƯƠNG 1: TỔNG QUAN**

Trong chương này, chúng tôi giới thiệu chung về báo cáo, động lực nghiên cứu, xác định mục tiêu và phạm vi tương ứng, từ đó nêu lên các đóng góp chính của báo cáo. Phần tóm tắt từng chương trong báo cáo được trình bày ở cuối chương này.

1.1. Giới thiệu chung

1.2. Động lực nghiên cứu

Đáp ứng yêu cầu môn học, …

Tìm hiểu kiến thức để ứng dụng trong tương lai, …

1.3. Phạm vi báo cáo

1.4. Thách thức

* Kiến thức mới, nhiều khái niệm chuyên sâu về hệ quản trị cơ sở dữ liệu mà chúng tôi lần đầu tiên tiếp cận.
* Lượng thông tin, tài liệu rất lớn từ cộng đồng phát triển cũng như là các nhà cung cấp hoặc xây dựng hệ quản trị cơ sở dữ liệu. Chúng tôi cần phải chọn lọc thông tin phù hợp.

1.5. Đóng góp của báo cáo

* Tìm hiểu về hệ quản trị cơ sở dữ liệu NoSQL, cụ thể là CouchDB.
* Cài đặt trên 2 máy trở lên và thực hiện truy vấn giữa hai máy.

1.6. Cấu trúc báo cáo

Báo cáo này này được trình bày trong 4 chương, nội dung chính được tóm tắt như dưới đây:

Chương 1: Tổng quan.

Chương 2: Hệ quản trị cơ sở dữ liệu CouchDB.

Chương 3: Cài đặt hai máy trở lên và thực hiện truy vấn giữa hai máy.

Chương 4: Thao tác dữ liệu qua lại giữa hai máy.

**CHƯƠNG 2: HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU COUCHDB**

1. **NoSQL là gì?**

* Cơ sở dữ liệu NoSQL là viết tắt của “Not Only SQL” hoặc “Not SQL”. Mặc dù một thuật ngữ tốt hơn NoREL NoSQL sẽ được sử dụng. Carl Strozz đã giới thiệu khái niệm NoSQL vào năm 1998.
* Cơ sở dữ liệu phi quan hệ được thiết kế đặc biệt cho các mô hình dữ liệu cụ thể và có các lược đồ linh hoạt để xây dựng các ứng dụng hiện đại. Chúng được công nhận rộng rãi vì chúng dễ phát triển, cả về chức năng và hiệu suất ở quy mô (Bhat & Jadhav, 2010). Sử dụng nhiều mô hình dữ liệu khác nhau, bao gồm tài liệu, đồ thị, giá trị khóa, tầm nhìn và tra cứu.
* Như chúng ta sẽ thấy, thông tin thường được tổ chức thông qua các tài liệu và nó rất hữu ích khi chúng ta không có sơ đồ chính xác về những gì sẽ được lưu trữ.
* Cơ sở dữ liệu NoSQL được sử dụng cho các kho dữ liệu phân tán với nhu cầu lưu trữ dữ liệu lớn. NoSQL được sử dụng cho dữ liệu lớn và ứng dụng web thời gian thực. Ví dụ, các công ty như Twitter, Facebook, Google thu thập hàng terabyte dữ liệu người dùng mỗi ngày.
* Hệ thống cơ sở dữ liệu NoSQL bao gồm một loạt các công nghệ cơ sở dữ liệu có thể lưu trữ dữ liệu có cấu trúc, bán cấu trúc, phi cấu trúc và đa hình.

1. **Giới thiệu về hệ quản trị cơ sở dữ liệu CouchDB**
2. **Lịch sử ra đời nguồn gốc**

**Logo, company name

Description automatically generated**

*Hình 1. Logo CouchDB*

*CouchDB* là từ viết tắt của cụm “*Cluster Of Unreliable Commodity Hardware Database*”. Dự án CouchDB được tạo ra vào tháng 4 năm 2005, ban đầu nó được viết bằng C++ bởi Damien Katz, cựu nhà phát triển Lotus Notes tại IBM. Anh ấy đã tự tài trợ cho dự án trong gần hai năm và phát hành nó dưới dạng một dự án mã nguồn mở theo General Public GNU*.*

CouchDB sau đó đã loại bỏ C++ và chuyển sang Erlang để phát triển vào tháng 2 năm 2006.

* CouchDB đã trở thành một dự án Apache Incubator và đã chuyển đổi giấy phép của nó thành Giấy phép Apache vào tháng 2 năm 2008.
* Phiên bản beta 0.10.0 của CouchDB được xuất bản vào tháng 10 năm 2009. Vài tháng sau, nó hoàn thành dự án cấp cao nhất. Điều này dẫn đến phiên bản ổn định đầu tiên được phát hành vào tháng 7 năm 2010.
* Đầu năm 2012, Katz rời dự án để tập trung vào Couchbase Server. Kể từ khi Katz rời đi, dự án vườn ươm Apache CouchDB vẫn tiếp tục, phát hành bản 1.2 vào tháng 4 năm 2012 và bản 1.3 vào tháng 4 năm 2013. Vào tháng 7 năm 2013, cộng đồng CouchDB đã hợp nhất cơ sở mã cho BigCouch , phiên bản CouchDB theo nhóm của Cloudant , vào dự án Apache. Khung phân cụm BigCouch được bao gồm trong bản phát hành hiện tại của Apache CouchDB.
* CouchDB đã xuất bản 1.2.0 vào tháng 4 năm 2012, 1.3.0 vào tháng 4 năm 2013, 1.4.0 vào tháng 8 năm 2013, 1.5.0 vào tháng 11 năm 2013, 1.6.0 vào tháng 6 năm 2014.
* Vào tháng 9 năm 2016, CouchDB đã xuất bản 2.0.0, tính năng chính của phiên bản này là nó hỗ trợ phân cụm, công việc này ban đầu được thực hiện trong dự án BigCouch và nó đã được hợp nhất vào phiên bản 2.0.0 của CouchDB.
* Phiên bản CouchDB là 2.3.0 vào tháng 12 năm 2018.
* Phiên bản mới nhất của CouchDB là 3.2.2 tháng 4 năm 2022.

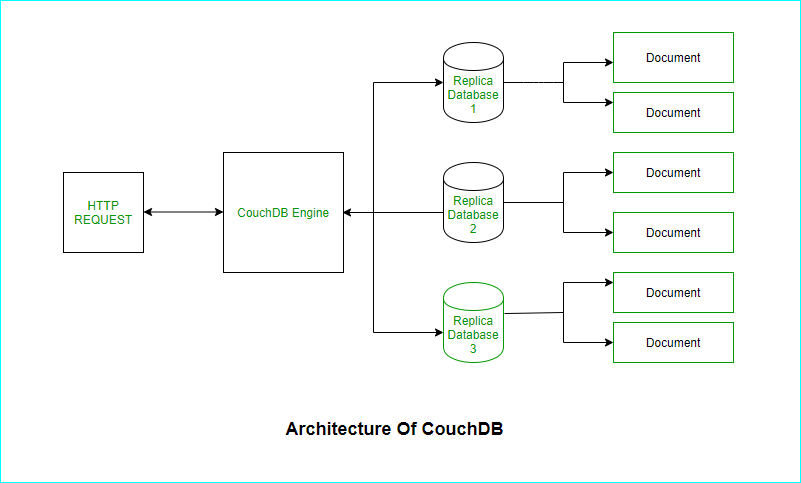
1. **Couchdb là gì?**

* Apache CouchDB là một cơ sở dữ liệu NoSQL hướng mã nguồn mở, được triển khai trong Erlang.
* CouchDB sử dụng nhiều định dạng và giao thức để lưu trữ, truyền và xử lý dữ liệu của nó. Nó sử dụng JSON để lưu trữ dữ liệu, JavaScript làm ngôn ngữ truy vấn của nó bằng cách sử dụng MapReduce và HTTP cho một API.
* CouchDB được thiết kế nhắm tới tính dễ sử dụng và phục vụ cho môi trường web.

1. **Lợi ích CouchDB mang lại?**

* CouchDB có API dạng RESTFul giúp cho việc giao tiếp với cơ sở dữ liệu được đơn giản.
* Các RESTFul API rất trực quan và dễ thao tác.
* Dữ liệu được lưu dưới cấu trúc document rất mềm dẻo, chúng ta không cần phải lo lắng về cấu trúc dữ liệu.
* Map/reduce giúp việc lọc, tìm, tổng hợp dữ liệu dễ hơn bao giờ hết.
* Nhân bản / đồng bộ là sức mạnh đặc biệt của CouchDB mà hiếm database nào có.

1. **Kiến trúc**



*Hình 2. Kiến trúc của CouchDB*

* **CouchDB Engine**: Nó dựa trên B – tree và trong đó, dữ liệu được truy cập bằng các khóa hoặc phạm vi khóa ánh xạ trực tiếp tới các hoạt động của B – tree bên dưới. Nó là cốt lõi của hệ thống quản lý để lưu trữ dữ liệu nội bộ, tài liệu và chế độ xem.
* **HTTP Request**: Nó được sử dụng để tạo chỉ mục và trích xuất dữ liệu từ tài liệu. Nó được viết bằng JavaScript cho phép tạo các chế độ xem Adhoc được tạo từ các công việc MapReduce.
* **Document**: Nó lưu trữ một lượng lớn dữ liệu.
* **Replica Database**: Nó được sử dụng để sao chép dữ liệu vào cơ sở dữ liệu cục bộ hoặc từ xa và đồng bộ hóa tài liệu thiết kế.

1. **Mô hình dữ liệu**

* Database là cấu trúc dữ liệu lớn nhất của CouchDB.
* Mỗi database là 1 danh sách các document độc lập.
* Document bao gồm dữ liệu người dùng thao tác lẫn thông tin về phiên bản của dữ liệu để tiện việc merge dữ liệu.
* CouchDB sử dụng cơ chế phiên bản hoá dữ liệu để tránh tình trạng khoá dữ liệu khi đang ghi.

1. **Các tính năng của CouchDB**

Các tính năng của CouchDB bao gồm:

* **Sao chép**: Nó cung cấp hình thức sao chép đơn giản nhất và không có cơ sở dữ liệu nào khác có thể sao chép đơn giản như vậy.
* **Lưu trữ tài liệu**: CouchDB là một NoSQL database dạng document. Document là một đơn vị dữ liệu (giống như 1 object của Javascript), mỗi field có một tên riêng không trùng nhau, chứa các loại dữ liệu như chữ, số, Boolean, danh sách…

CouchDB cung cấp 1 RESTFul API cho việc đọc và ghi (thêm, sửa, xoá) document.

Sau đây là 1 ví dụ về 1 document:

| { |  |
| --- | --- |
|  | “carid”: “1” |
|  | “carname”: “Mirage” |
|  | “vendor”: “Mitsubishi” |
|  | “price”: 24500 |
| } |  |

* **Thuộc tính ACID**: Khi dữ liệu được ghi xuống ổ cứng thì nó sẽ không bị ghi đè. Bất kì thay đổi nào (thêm, sửa, xoá) đều theo chuẩn Atomic, có nghĩa là dữ liệu sẽ được lưu lại toàn diện hoặc không được lưu lại. Database không bao giờ thêm hay sửa một phần dữ liệu. Hầu hết các cập nhật đều được serialized để đảm bảo tất cả người dùng có thể đọc document mà không bị chờ đợi hoặc gián đoạn.
* **Bảo mật**: Nó cũng cung cấp bảo mật cấp cơ sở dữ liệu và các quyền được chia thành người đọc và quản trị viên nơi người đọc có thể thực hiện cả việc đọc và ghi vào cơ sở dữ liệu.
* **Map/Reduce**: Lý do chính cho sự phổ biến của CouchDB là một hệ thống bản đồ/thu nhỏ.
* **Xác thực**: CouchDB hỗ trợ bạn duy trì xác thực mở thông qua cookie phiên giống như một ứng dụng web.
* **Được xây dựng cho Ngoại tuyến**: CouchDB có thể sao chép sang các thiết bị như điện thoại thông minh có tính năng chuyển sang ngoại tuyến và xử lý đồng bộ hóa dữ liệu cho bạn khi thiết bị trực tuyến trở lại.
* **Khả năng nén (Compaction)**: Nén là 1 hành động giúp giải phóng dung lượng ổ cứng được sử dụng bằng cách xoá đi các dữ liệu không còn được sử dụng. Khi tiến hành nén dữ liệu ở 1 file thì 1 file mới với định dạng. Compaction sẽ được tạo ra và dữ liệu sẽ được sao chép vào file mới này.

Khi quá trình copy hoàn thành thì file cũ sẽ được xoá bỏ. Database vẫn online trong quá trình nén và các thao tác thay đổi / đọc dữ liệu vẫn diễn ra bình thường.

* **Tính nhất quán cuối cùng**: CouchDB đảm bảo tính nhất quán cuối cùng để cung cấp cả tính khả dụng và dung sai phân vùng.
* **API HTTP**: Tất cả các mục có một URI duy nhất (Mã định danh tài nguyên duy nhất) được hiển thị qua HTTP. Nó sử dụng các phương thức HTTP như POST, GET, PUT và DELETE cho bốn thao tác CRUD (Tạo, Đọc, Cập nhật, Xóa) cơ bản trên tất cả các tài nguyên.
* **Views:** Views là một tập hợp con của database và dựa trên câu truy vấn được chạy trên một hoặc nhiều bảng database.

Không giống như cơ sở dữ liệu SQL, nơi dữ liệu phải được phân tách cẩn thận thành các bảng, dữ liệu trong CouchDB được lưu trữ trong các tài liệu bán cấu trúc. Các tài liệu CouchDB rất linh hoạt và mỗi tài liệu có cấu trúc ngầm định riêng, giúp giảm bớt các vấn đề và cạm bẫy khó khăn nhất của các lược đồ bảng sao chép hai chiều và dữ liệu chứa trong chúng.

Dữ liệu trong CouchDB được lưu trữ trong các document. Bạn có thể tưởng tượng như 1 database là 1 table và 1 document là 1 row. Khi chúng ta muốn trình bày dữ liệu bằng nhiều góc nhìn khác nhau thì chúng ta cần 1 phương pháp để filter, tổ chức để hiển thị kết quả cuối cùng.

Để giải quyết vấn đề cấu trúc trở lại dữ liệu phi cấu trúc và bán cấu trúc, CouchDB tích hợp một mô hình views. Views là phương pháp tổng hợp và báo cáo về các tài liệu trong cơ sở dữ liệu và được xây dựng theo yêu cầu để tổng hợp, tham gia và báo cáo về các tài liệu cơ sở dữ liệu. Vì các dạng xem được xây dựng động và không ảnh hưởng đến tài liệu bên dưới, nên bạn có thể có nhiều dạng biểu diễn dạng xem khác nhau của cùng một dữ liệu tùy thích.

CouchDB định nghĩa các view là các hàm Map-Reduce, sử dụng các view đó để lấy dữ liệu, không tự chạy hàm JavaScript. Thay vào đó, khi bạn truy vấn chế độ view của mình , CouchDB sẽ lấy source code và chạy nó cho bạn trên mọi document trong database mà view đã được xác định.

* **Lợi ích của view:**

Lọc các document trong cơ sở dữ liệu để tìm những document liên quan đến một quy trình cụ thể.

Trích xuất dữ liệu từ tài liệu của bạn và trình bày nó theo một thứ tự cụ thể.

Xây dựng các chỉ mục hiệu quả để tìm tài liệu theo bất kỳ giá trị hoặc cấu trúc nào nằm trong đó.

Sử dụng các chỉ mục này để thể hiện mối quan hệ giữa các tài liệu.

Với views, ta có thể thực hiện tất cả các loại tính toán trên dữ liệu trong document của mình.

Ví dụ: nếu các document trình bày các giao dịch tài chính của công ty bạn, thì view có thể trình bày cho ta thấy về chi tiêu trong tuần, tháng hoặc năm trước.

1. **Efficient Lookups**

B-trees sao lưu kết quả dạng xem được sắp xếp theo khóa chỉ được tạo một lần, khi truy vấn một dạng xem lần đầu tiên và tất cả các truy vấn tiếp theo sẽ chỉ đọc cây B thay vì thực hiện lại chức năng bản đồ cho tất cả các tài liệu. Tuy nhiên, điều gì sẽ xảy ra khi bạn thay đổi tài liệu, thêm tài liệu mới hoặc xóa tài liệu.

CouchDB sẽ tìm các hàng trong kết quả view được tạo bởi một document cụ thể. Nó đánh dấu những hàng này không hợp lệ để không còn hiển thị trong view.

Nếu document bị xóa, kết quả B-Tree phản ánh trạng thái của cơ sở dữ liệu.

Nếu một document được cập nhật, document mới sẽ được chạy qua hàm map và kết quả là các dòng mới được chèn vào B-Tree tại các vị trí đó.

* + Find one
  + Find many

1. **Ưu và nhược điểm CouchDB**
2. **Ưu điểm của CouchDB**

* API HTTP được sử dụng để giao tiếp dễ dàng.
* Các RESTFul API rất trực quan và dễ thao tác.
* Nó được sử dụng để lưu trữ bất kỳ loại dữ liệu nào.
* Dữ liệu được lưu dưới cấu trúc document rất linh hoạt, chúng ta không cần phải lo lắng về cấu trúc dữ liệu.
* Map/reducecho phép tối ưu hóa việc kết hợp dữ liệu.
* Cấu trúc của CouchDB rất đơn giản
* Lập chỉ mục và truy xuất nhanh.
* Nhân bản / đồng bộ là sức mạnh đặc biệt của CouchDB.

1. **Nhược điểm của CouchDB**

* CouchDB chiếm một không gian lớn cho chi phí hoạt động, đây là một bất lợi lớn so với các cơ sở dữ liệu khác.
* Truy vấn tùy ý là tốn kém.
* Có một chút không gian bổ sung trên đầu với CouchDB so với hầu hết các lựa chọn thay thế.
* Chế độ xem tạm thời trên bộ dữ liệu khổng lồ rất chậm.
* Nó không hỗ trợ giao dịch
* Sao chép cơ sở dữ liệu lớn có thể thất bại.

1. So sánh CouchDB và MongoDB

| STT | Yếu tố | CouchDB | MongoDB |
| --- | --- | --- | --- |
| 1 | Lược đồ lưu trữ | Dữ liệu được lưu trữ ở định dạng JSON. Cơ sở dữ liệu CouchDB sử dụng mô hình kiểu hướng tài liệu. | Dữ liệu được lưu trữ ở định dạng BSON và tuân theo mô hình loại hướng tài liệu. |
| 2 | Ngôn ngữ lập trình | CouchDB sử dụng ngôn ngữ lập trình Erlang để phát triển. | MongoDB sử dụng ngôn ngữ lập trình C++ để phát triển. |
| 3 | Hỗ trợ cho SQL | Không hỗ trợ SQL trong cơ sở dữ liệu CouchDB. | MongoDB hỗ trợ các truy vấn SQL (chỉ đọc) sử dụng trình kết nối MongoDB. |
| 4 | Sử dụng giao diện | Giao diện được sử dụng bởi cơ sở dữ liệu CouchDB là giao diện HTTP/REST. Giao diện được sử dụng được xác định đúng cho người dùng để có thể dễ dàng sử dụng. | MongoDB sử dụng các giao thức khác nhau thay cho giao diện. Các giao thức được sử dụng là giao thức tùy chỉnh và giao thức nhị phân nằm trên mô hình TCP/IP. |
| 5 | Lưu trữ | CouchDB sử dụng documents để lưu trữ dữ liệu trong cơ sở dữ liệu. | MongoDB sử dụng collections để lưu trữ dữ liệu trong database. Và trong collections, có documents ở đó để lưu trữ dữ liệu. |
| 6 | Mô hình nhân bản | Mô hình nhân bản được hỗ trợ bởi mô hình cơ sở dữ liệu CouchDB là mô hình master-master | Mô hình nhân bản được hỗ trợ bởi mô hình cơ sở dữ liệu MongoDB là mô hình master-slave. |
| 7 | Phương thức truy vấn | Phương thức truy vấn loại map/reduce được sử dụng trong mô hình cơ sở dữ liệu CouchDB | Ngôn ngữ truy vấn hướng đối tượng được sử dụng và phương thức truy vấn map/redure cũng được đưa ra trong mô hình cơ sở dữ liệu MongoDB. |
| 8 | Hiệu suất | Hiệu suất CouchDB tốt hơn nhiều so với lược đồ cơ sở dữ liệu MongoDB. | Cơ sở dữ liệu MongoDB phải đối mặt với vấn đề về hiệu năng và thua kém khi so sánh với cơ sở dữ liệu CouchDB |
| 9 | Cơ chế chuyển đổi dự phòng | Quy trình chuyển đổi dự phòng chậm trong cơ sở dữ liệu CouchDB. Khi một bản gốc ngừng hoạt động, cần có thời gian để chuyển sang bản chính tiếp theo cho các hoạt động lưu trữ dữ liệu. | Trong cơ sở dữ liệu MongoDB, cơ chế chuyển đổi dự phòng nhanh hơn so với cơ sở dữ liệu CouchDB. |

1. **Tạo và thêm mới dữ liệu**
2. **Hướng dẫn cài đặt trong Apache CouchDB**

B1: Thực hiện tải file apache CouchDB 3.2.2

Graphical user interface

Description automatically generated with medium confidence

B2: Mở file apache-couchdb-3.2.2

Graphical user interface, text, application

Description automatically generated

B3: Chọn “Next”

Graphical user interface, text, application, email

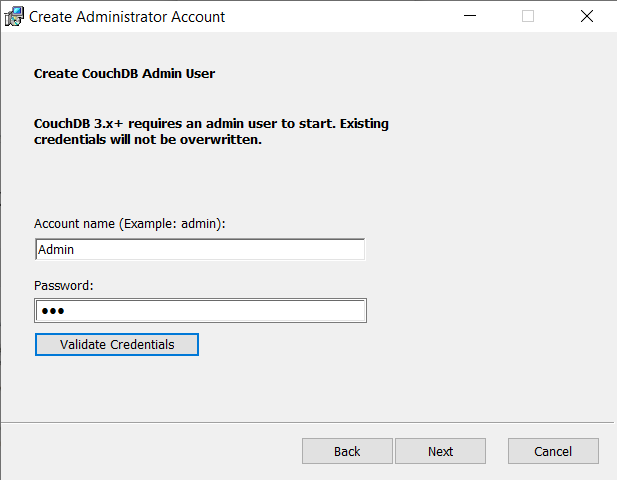
Description automatically generated

B4: Chọn “I accept the terms in the License Agreement” và nhấn “Next”

Graphical user interface, text, application

Description automatically generated

B5: Chọn nơi lưu và nhấn “Next”



B6: Tạo account và password, nhấn “Next”

Graphical user interface, text, application, email

Description automatically generated

B7: Set Cookie value và nhấn “Next”

Graphical user interface, text, application, email

Description automatically generated

B8: Chọn “Install” để tiến hành cài đặt

Graphical user interface, text, application

Description automatically generated

B9: Chọn “Finish”

1. **Thao tác document bằng tiện ích cURL**
2. **Tạo cơ sở dữ liệu bằng cURL**

Sử dụng cú pháp sau để tạo cơ sở dữ liệu trong CouchDB bằng cách gửi yêu cầu HTTP đến máy chủ bằng phương thức PUT thông qua tiện ích cURL.

**Syntax**: curl -X PUT http://username:password@localhost:5984/database\_name

**Ví dụ**: Tạo một cơ sở dữ liệu tên “car\_sales”. Khi cơ sở dữ liệu của bạn được tạo thành công, máy chủ sẽ trả lại cho bạn phản hồi ở dạng tài liệu JSON với nội dung {“ok”:true}.

Text

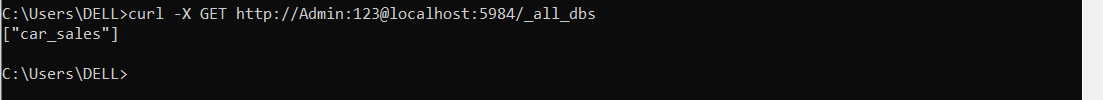
Description automatically generated

*Hình 4. Cách tạo database bằng cURL*

Bạn có thể xác minh rằng nếu cơ sở dữ liệu đã được tạo bằng cách liệt kê tất cả các cơ sở dữ liệu bằng lệnh sau:

**Syntax**: curl -X GET http://username:password@localhost:5984/\_all\_dbs

Ví dụ:



*Hình 5. Cú pháp kiểm tra các database đã tồn tại*

1. **Xóa cơ sở dữ liệu bằng cURL**

Syntax: curl -X DELETE <http://username:password@localhost:5984/database_name>

Ví dụ:



1. **Tạo document bằng cURL**

* Sử dụng curl bằng Command prompt:
* **Thêm 1 document**

Syntax: curl -X POST <http://username:password@localhost:5984/database_name/>

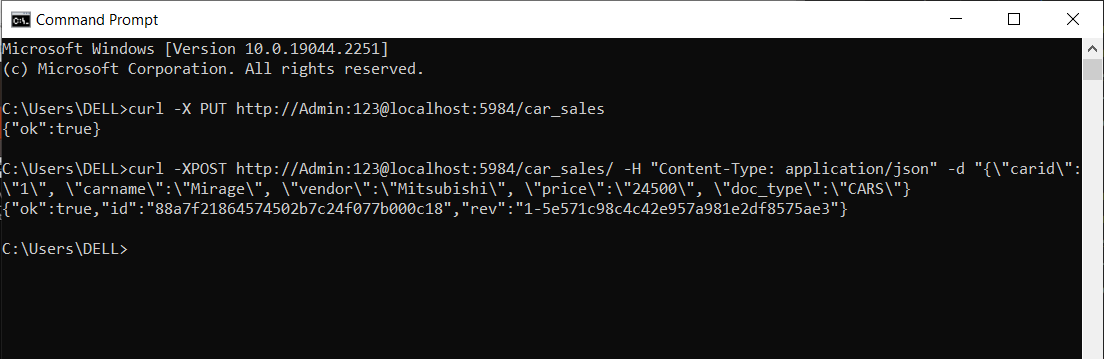
-H “Content-Type: application/json” -d “data ở dạng JSON”.

Trong đó:

* Curl -X POST: đang muốn gửi yêu cầu đến cURL rằng muốn thêm một số dữ liệu lên máy chủ.
* -H “Content-Type: application/json”: mô tả loại dữ liệu muốn gửi.
* -d “…” là dữ liệu dạng JSON muốn thêm vào cơ sở dữ liệu.

Lưu ý: trong Windows, phải thay thế các dấu ngoặc kép trong JSOn bằng dấu gạch chéo ngược.

VD: {“name”:”Nhựt”} -> {\”name\”:\”Nhựt\”}



*Hình 5. Cách tạo 1 document bằng cURL*

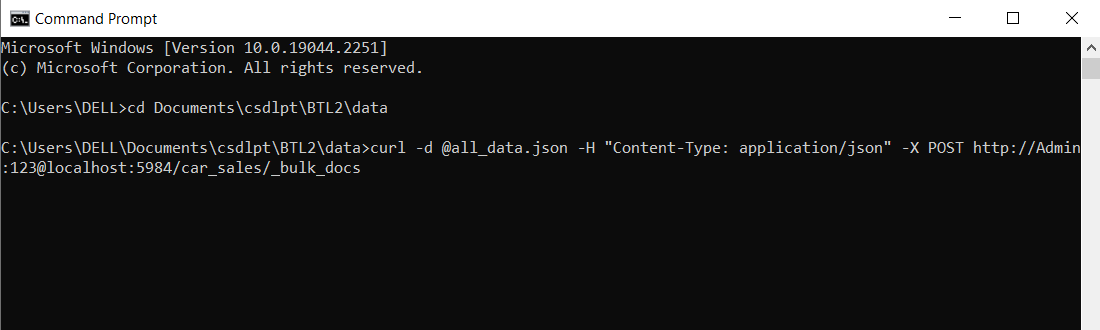
Phản hồi của CouchDB đối với yêu cầu đến HTTP trên chứa 3 trường:

* “ok”: cho biết thao tác được thực hiện thành công.
* “id”: lưu trữ id của documents
* “rev”: cho biết id bản sửa đổi. Mỗi khi bạn sửa đổi (cập nhật hoặc sửa đổi) một tài liệu, giá trị \_rev sẽ được CouchDB tạo ra. Nếu bạn muốn cập nhật hoặc xóa một tài liệu, CouchDB mong bạn đưa vào trường \_rev của bản sửa đổi mà bạn muốn thay đổi. Khi CouchDB chấp nhận thay đổi, nó sẽ tạo ra một số sửa đổi mới. Cơ chế này đảm bảo kiểm soát đồng thời.
* **Thêm dữ liệu hàng loạt:**

Mỗi khi một tài liệu được lưu trữ hoặc cập nhật trong CouchDB, B-tree bên trong sẽ được cập nhật. Chèn hàng loạt giúp tăng hiệu quả cả về không gian lưu trữ và thời gian, bằng cách hợp nhất nhiều bản cập nhật cho các nút B-tree trung gian.

Để chèn hàng loạt tài liệu vào cơ sở dữ liệu, bạn cần cung cấp cấu trúc JSON với mảng tài liệu mà bạn muốn thêm vào cơ sở dữ liệu. Bạn có thể bao gồm ID tài liệu hoặc cho phép ID tài liệu được tạo tự động.

**Syntax:** curl -d @Database\_file\_name.json -H "Content-type: application/json" -X POST <http://username:password@localhost:5984/Database_name/_bulk_docs>



*Hình 6. Cách thêm dữ liệu hàng loạt trong CouchDB*

Kết quả:

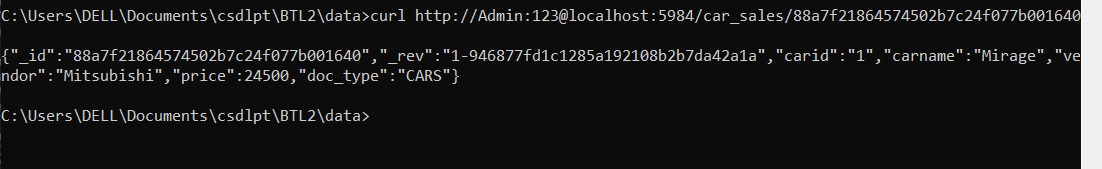
Text

Description automatically generated

*Hình 7. Máy chủ phản hồi thành công sau khi thực thi lệnh thêm dữ liệu hàng loạt.*

* Đoc dữ liệu: để đọc tài liệu bằng cách sử dụng curl, chúng ta cần biết giá trị id của id tài liệu.

Syntax: curl <http://username:password@localhost:5984/database_name/_id>



*Hình 8. Cách xem document đã được tạo thông qua id*

1. **Cập nhật dữ liệu**

* Trường rev trong CouchDB là một thứ đóng vai trò quan trọng trong cách CouchDB xử lý việc giải quyết các bản cập nhật xung đột có thể xảy ra.
* Nếu bạn muốn cập nhật hoặc xóa một tài liệu, CouchDB yêu cầu bạn bao gồm trường \_rev của bản sửa đổi mà bạn muốn thay đổi. Khi CouchDB chấp nhận thay đổi, nó sẽ tạo một số sửa đổi mới. Cơ chế này đảm bảo rằng, trong trường hợp ai đó thực hiện thay đổi mà bạn không biết trước khi bạn yêu cầu cập nhật tài liệu, CouchDB sẽ không chấp nhận bản cập nhật của bạn vì bạn có khả năng ghi đè lên dữ liệu mà bạn không biết là đã tồn tại.
* Để cập nhật tài liệu, cần phải chỉ định cả \_id và \_rev của tài liệu.

**Syntax**: curl -X PUT http://username:password@localhost:5984/database\_name/document\_id/ -d '{ "field" : "value", "\_rev" : "revision id" }'

Trước hết, truy xuất id sửa đổi của tài liệu sẽ được cập nhật. Bạn có thể tìm \_rev của tài liệu trong tài liệu bằng cách sử dụng lệnh sau:

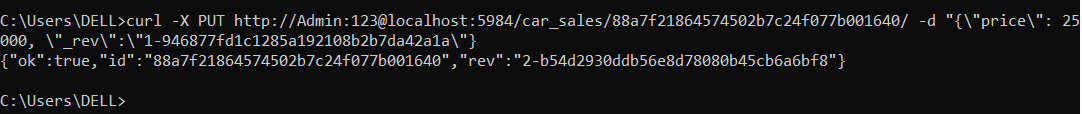
curl -X GET http://username:password@localhost:5984/database\_name /document\_id

Text

Description automatically generated

*Hình 9: Truy vấn document trước khi thao tác cập nhật*

Ví dụ: Cập nhật giá (price) của xe Mirage từ 24500 thành 25000.



Ta có thể thấy trường \_rev đã có giá trị mới.

1. **Xóa dữ liệu**

* Để thực hiện xóa tài liệu chúng ta cần chỉ định cả \_id và rev của tài liệu.
* Sau khi thực hiện xóa, CouchDB đã gán một giá trị rev cho việc xóa. Tại sao chúng ta lại có một giá trị rev cho một tài liệu đã xóa?
* Tại thời điểm này, tài liệu chưa thực sự bị xóa hoàn toàn khỏi cơ sở dữ liệu, nó chỉ được thay thế bằng một tài liệu trống đã được đánh dấu là đã xóa.

Syntax: curl -X DELETE <http://username:password@localhost:5984/_id> -H “If-Match: \_rev”

1. **Thao tác document bằng Fauxton**

* Futon là giao diện quản trị cài sẵn, dựa trên web của CouchDB. Nó cung cấp một giao diện đồ họa đơn giản mà bạn có thể tương tác với CouchDB. Đó là một giao diện ngây thơ và nó cung cấp quyền truy cập đầy đủ vào tất cả các tính năng của CouchDB. Sau đây là danh sách các tính năng đó:

1. **Tạo cơ sở dữ liệu bằng Fauxton**

B1: Mở liên kết sau trong trình duyệt web: <http://127.0.0.1:5984/_utils/> để đăng nhập vào CouchDB.

Graphical user interface, text, application, email

Description automatically generated

B2: Nhấn vào tab "Create database" để tiến hành tạo cơ sở dữ liệu.

Graphical user interface, text, application, email

Description automatically generated

B3: Cơ sở dữ liệu “car\_sales” được tạo thành công.

Graphical user interface, text, application, email

Description automatically generated

1. **Tạo document bằng Fauxton**

B1: Nhấn vào cơ sở dữ liệu muốn tạo document

Graphical user interface, text, application

Description automatically generated

B2: Chọn “Create Document” để thêm 1 document

Shape, rectangle

Description automatically generated

B3: Nhập dữ liệu cần tạo document dạng JSON

Graphical user interface, text, application

Description automatically generated

B4: Chọn “Create Document” và xem dữ liệu ở nhiều dạng khác nhau.

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

1. **Cập nhật document bằng Fauxton**

Bạn cũng có thể cập nhật/thay đổi/chỉnh sửa tài liệu của mình sau khi tạo. Bấm vào tùy chọn chỉnh sửa (hình cây bút).

Text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Thực hiện cập nhật “price” từ 24500 thành 25000

Graphical user interface, text, application

Description automatically generated

Chọn “Save Changes” để tiến hành cập nhật document

Graphical user interface, text

Description automatically generated

1. Xóa document bằng Fauxton

Tích vào document cần xóa và nhấn vào biểu tượng xóa trên màn hình.

Text

Description automatically generated with medium confidence

Hiển thị thông báo bạn có chắc chắn muốn xóa dữ liệu không? Chọn “OK” để thực hiện xóa.

Graphical user interface, text, application

Description automatically generated

Xóa document thành công.

Graphical user interface, application

Description automatically generated

**CHƯƠNG 3: CÀI ĐẶT HAI MÁY TRỞ LÊN VÀ THỰC HIỆN TRUY VẤN GIỮA HAI MÁY**

1. Thiết lập kết nối giữa hai máy

Tắt tường lửa ở cả 2 máy:

Text, application, email

Description automatically generated

Kết nối 2 máy bằng Radmin VPN:

Địa chỉ IP máy 1:

Graphical user interface, text, application

Description automatically generated

Địa chỉ IP máy 2:

Graphical user interface, text

Description automatically generated

Máy 1 tạo network (CouchDb\_network):

A screenshot of a computer

Description automatically generated with medium confidence

Máy 2 kết nối vào network:

A screenshot of a computer screen

Description automatically generated with medium confidence

Đổi CouchDb bind\_address thành 0.0.0.0 trên 2 máy (mặc định là 127.0.0.1):

Cách 1: Thay đổi thuộc tính bind\_address ở tab Configuration

Graphical user interface, application

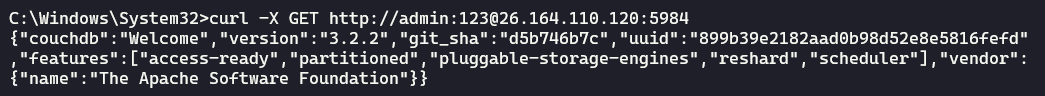
Description automatically generated

Cách 2: Sử dụng lệnh dưới đây trong Cmd

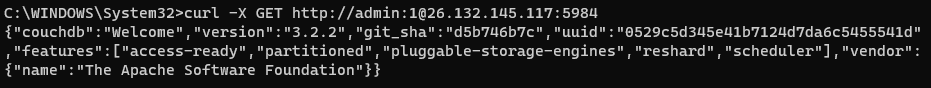
curl -X PUT http://<admin\_user>:<password>@localhost:5984/\_node/\_local/\_config/chttpd/bind\_address -d "\"0.0.0.0\"

Kiểm tra kết nối:

Máy 1 ping đến cổng 5984 của CouchDb bên máy 2 bằng tài khoản admin, pass 123 của máy 2:



Máy 2 ping đến cổng 5984 của CouchDb bên máy 1 bằng tài khoản admin, pass 1 của máy 1:



Output như trên là ping thành công!

Phần 2: Tạo user và phân quyền

// Export data from couchdb

curl -X GET http://127.0.0.1:5984/[mydatabase]/\_all\_docs\?include\_docs\=true > /Desktop/db.json

// Import data to couchdb

curl -d @db.json -H "Content-type: application/json" -X POST http://127.0.0.1:5984/[mydatabase]/\_bulk\_docs

// Replicate couchdb data

curl -H 'Content-Type: application/json' -X POST http://localhost:5984/\_replicate -d ' {"source": "http://example.com:5984/dbname/", "target": "http://localhost@:5984/dbname/"}'

// Create User

curl -X PUT http://localhost:5984/\_users/org.couchdb.user:jan \

-H "Accept: application/json" \

-H "Content-Type: application/json" \

-d '{"name": "jan", "password": "apple", "roles": [], "type": "user"}'

// Permission

curl -X PUT http://localhost:5984/mydatabase/\_security \

-u anna:secret \

-H "Content-Type: application/json" \

-d '{"admins": { "names": [], "roles": [] }, "members": { "names": ["jan"], "roles": [] } }'

// Check

curl -X GET http://<ip>:5984/[db]

// Query

curl -X POST http://<ip>:5984/[db]/\_find \

-H "Content-Type: application/json" \

-d \

{

"selector": {

"year": {"$gt": 2010}

},

"fields": ["\_id", "\_rev", "year", "title"],

"sort": [{"year": "asc"}],

"limit": 2,

"skip": 0,

"execution\_stats": true

}

// Insert

curl -H "Content-type: application/json" -X POST http://127.0.0.1:5984/[mydatabase]/\_bulk\_docs \

-d

{

“Docs”: [

{

“Key”: “value”

}

]

}

// Update

curl -X PUT http://localhost:5984/[db]/document\_id/ -d ‘{"field": "value", "\_rev": "revision id"}'

**CHƯƠNG 4: THAO TÁC DỮ LIỆU QUA LẠI GIỮA HAI MÁY**

1. Thêm, xóa, sửa qua lại giữa hai máy
2. Cơ chế nhân bản trong phân tán trong CouchDB