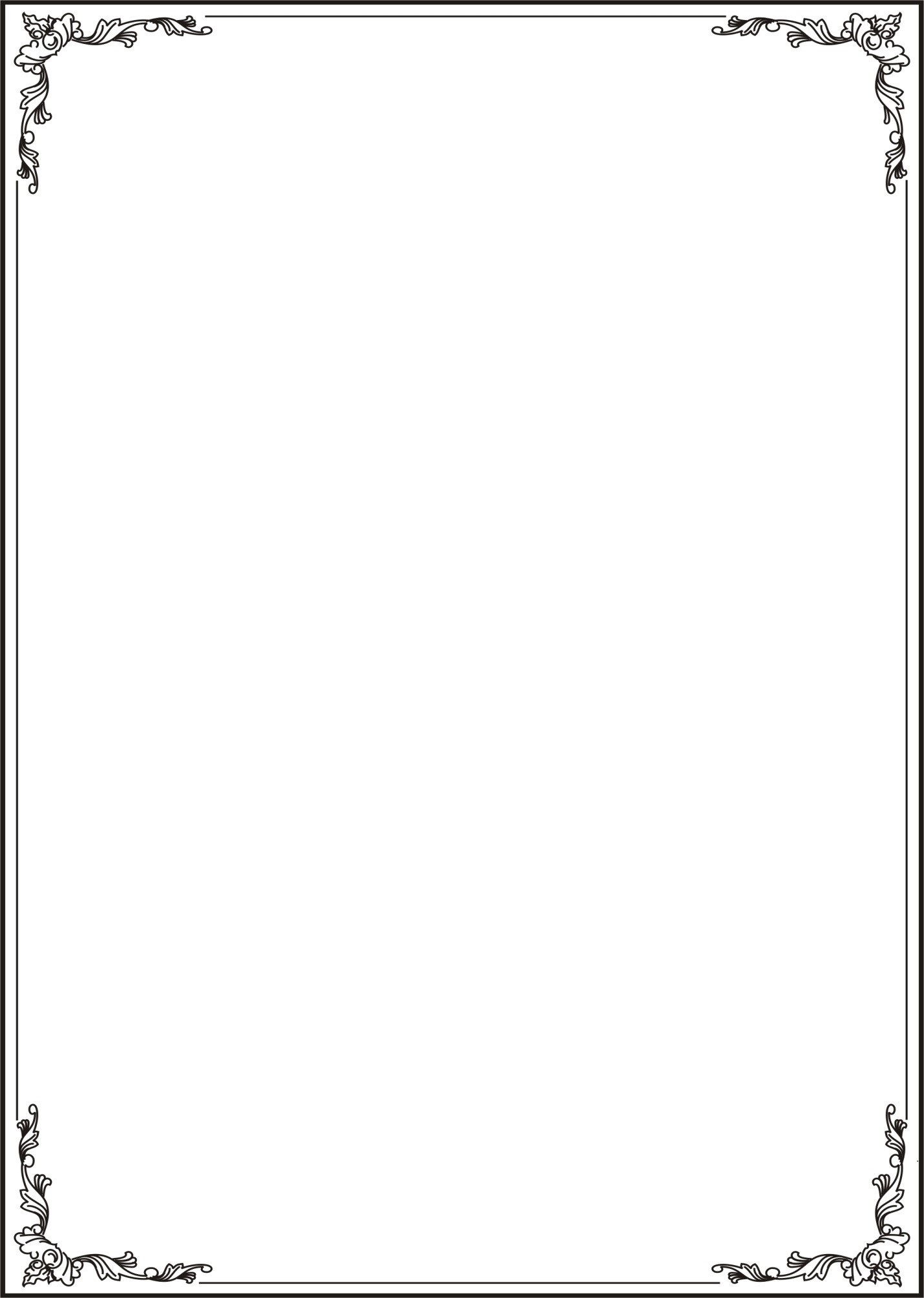
**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM TP.HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

****-----🙞🙜🕮🙞🙜-----

**BÁO CÁO ĐỒ ÁN CUỐI KÌ**

**NHÓM 9**

**Dynamo DB**

**Giảng viên hướng dẫn: Thầy Lương Trần Hy Hiến**

**Sinh viên thực hiện: Nguyễn Thiên Ân – 43.01.104.003**

**Trần Bá Khoa – 43.01.104.081**

**Nguyễn Phước Nhã - 43.01.104.118**

**Vương Duy Tài – 43.01.104.153**

***Tp.Hồ Chí Minh, tháng 11 năm 2019***

MỤC LỤC

Lời mở đầu 2

I/ Giới thiệu chung 3

1. NoSQL là gì? 3

2. Cơ sở dữ liệu SQL so với cơ sở dữ liệu NoSQL 3

II/ Tổng quan về DynamoDB 5

1. Lợi ích 5

2. Ứng dụng 6

3. Trường hợp sử dụng 7

4. Các thương hiệu lớn sử dụng DynamoDB 9

III/ Môi trường làm việc với DynamoDB 11

1. Cài đặt DynamoDB 11

2. Kết nối vào DynamoDB 11

IV/ Tìm hiểu về DynamoDB 15

1. Các Components trong DynamoDB 15

2. DynamoDB API 18

3. Cách đặt tên và kiểu dữ liệu 20

a. Cách đặt tên 20

b. Kiểu dữ liệu 21

V/ Lập trình DynamoDB với ngôn ngữ JavaScript 25

1. Tính năng 25

2. Demo 25

VI/ Tài liệu tham khảo 32

**Lời Mở Đầu**

Amazon DynamoDB là cơ sở dữ liệu về khóa-giá trị và tài liệu, với hiệu năng hoạt động chỉ trong vài mili giây ở mọi quy mô. Đây là một cơ sở dữ liệu bền vững, nhiều bản chính, đa vùng, được quản lý toàn phần, có tích hợp tính năng bảo mật, sao lưu, khôi phục và lưu vào bộ nhớ đệm cho các ứng dụng trên quy mô Internet. DynamoDB có thể xử lý hơn 10 nghìn tỷ yêu cầu/ngày và hỗ trợ tối đa hơn 20 triệu yêu cầu/giây.

Nhiều doanh nghiệp phát triển nhanh nhất thế giới như Lyft, Airbnb, Redfin, Samsung, Toyota và Capital One dựa vào quy mô và hiệu năng của DynamoDB để hỗ trợ khối lượng công việc then chốt của họ.

Hàng trăm nghìn khách hàng AWS đã lựa chọn DynamoDB làm cơ sở dữ liệu về khóa-giá trị và tài liệu cho các ứng dụng di động, web, trò chơi, công nghệ quảng cáo, IoT và các ứng dụng khác cần quyền truy cập dữ liệu với độ trễ thấp ở mọi quy mô. Tạo một bảng mới cho ứng dụng của bạn và để DynamoDB đảm nhiệm phần còn lại.

I/ Giới thiệu chung

DynamoDB là môt dịch vụ quản lý NoSQL có khả năng đáp ứng hiệu suất cao và nhanh kèm theo khả năng mở rộng. Nếu bạn là một nhà phát triển, bạn có thể sử dụng DynamoDB để tạo ra một bảng có khả năng lưu trữ và truy xuất bất kỳ số lượng dữ liệu, mà vẫn có thể phục vụ cho bất kỳ mức độ request traffic.

DynamoDB tự động phân tán dữ liệu và traffic của một bảng ra một số lượng server vừa đủ để có thể xử lý request capacity đặt ra bởi khách hàng và lượng dữ liệu lưu trữ, và đồng thời đảm bảo hiệu suất nhanh và đồng nhất. Tất cả dự liệu được lưu trữ trên SSD và tự động được sao chép ra các vùng sẵn sằng (Availability Zones) trong một khu vực (Region) để cung cấp độ sẵn sàng cao và độ bền của dữ liệu (high availablity and data durability)

Nếu bạn là một nhà quản lý dữ liệu, bạn có thể tạo một bảng dữ liệu mới, mở rộng hay thu hẹp request capacity mà không bị giảm hiệu suất, và có thể thây được các thông số qua AWS Management Console. Với DynamoDB, bạn có thể phó thác gánh nặng quản lý và mở rộng dữ liệu cho AWS và không phải lo lắng về việc cung cấp hardware, thiết lập và cài đặt, sao chép dữ liệu.

1. NoSQL là gì ?

Cơ sở dữ liệu NoSQL là Cơ sở dữ liệu được xây dựng dành riêng cho mô hình dữ liệu và có sơ đồ linh hoạt để xây dựng các ứng dụng hiện đại. Cơ sở dữ liệu NoSQL được công nhận rộng rãi vì khả năng dễ phát triển, chức năng cũng như hiệu năng ở quy mô lớn. Các Cơ sở dữ liệu này sử dụng nhiều mô hình dữ liệu đa dạng, trong đó có văn bản, đồ thị, khóa – giá trị, trong bộ nhớ và tìm kiếm.

1. Cơ sở dữ liệu SQL so với Cơ sở dữ liệu NoSQL

Trong nhiều thập kỷ, mô hình dữ liệu chiếm ưu thế về lượng sử dụng để phát triển ứng dụng là mô hình dữ liệu quan hệ được sử dụng trong các cơ sở dữ liệu quan hệ, ví dụ như Oracle, DB2, SQL Server, MySQL và PostgreSQL. Mãi cho đến cuối những năm 2000, các mô hình dữ liệu khác mới bắt đầu được đưa vào áp dụng và sử dụng nhiều hơn. Để phân biệt và phân loại các cơ sở dữ liệu và mô hình dữ liệu mới này, từ "NoSQL" đã được đặt ra. Thông thường, thuật ngữ "NoSQL" được sử dụng tương đương với "phi quan hệ".

Mặc dù có nhiều loại cơ sở dữ liệu NoSQL với nhiều tính năng đa dạng, bảng sau sẽ minh họa một số khác biệt giữa cơ sở dữ liệu SQL và NoSQL.

|  |  |  |
| --- | --- | --- |
|  | Cơ sở dữ liệu quan hệ | Cơ sở dữ liệu NoSQL |
| Khối lượng công việc tối ưu | Cơ sở dữ liệu quan hệ được thiết kế dành cho các ứng dụng xử lý giao dịch trực tuyến (OLTP) trong giao dịch có độ ổn định cao và thích hợp để xử lí phân tích trực tuyến (OLAP). | Các cơ sở dữ liệu NoSQL trong bộ nhớ, giá trị–khóa, văn bản và đồ thị được thiết kế cho OLTP cho một số mẫu truy cập dữ liệu, bao gồm các ứng dụng có độ trễ thấp. Cơ sở dữ liệu tìm kiếm NoSQL được thiết kế để phục vụ phân tích dữ liệu có cấu trúc chưa hoàn chỉnh. |
| Mô hình dữ liệu | Mô hình quan hệ chuẩn hóa dữ liệu vào bảng được hình thành từ hàng và cột. Sơ đồ quy định rõ ràng bảng, hàng, cột, chỉ mục, mối quan hệ giữa các bảng và các thành tố cơ sở dữ liệu khác. Cơ sở dữ liệu sẽ thực thi tính toàn vẹn tham chiếu trong mối quan hệ giữa các bảng. | Cơ sở dữ liệu NoSQL cung cấp nhiều mô hình dữ liệu đa dạng, trong đó có văn bản, đồ thị, khóa–giá trị, trong bộ nhớ và tìm kiếm. |
| Thuộc tính ACID | Cơ sở dữ liệu quan hệ có các thuộc tính mang tính nguyên tố, nhất quán, tách biệt và bền vững (ACID):   * Tính nguyên tố đòi hỏi giao dịch phải được thực thi đầy đủ hoặc hoàn toàn không thực hiện. * Tính nhất quán đòi hỏi rằng khi giao dịch được thực hiện, dữ liệu phải nhất quán với sơ đồ cơ sở dữ liệu. * Tính tách biệt đòi hỏi rằng các giao dịch diễn ra đồng thời phải được thực thi tách biệt với nhau. * Tính bền vững đòi hỏi phải có khả năng phục hồi từ tình trạng hư hỏng hệ thống hoặc mất điện đột ngột về trạng thái đã biết cuối cùng. | Cơ sở dữ liệu NoSQL thường phải đánh đổi bằng cách nới lỏng một số thuộc tính ACID này của cơ sở dữ liệu quan hệ để có mô hình dữ liệu linh hoạt hơn có khả năng thay đổi quy mô theo chiều ngang. Việc này biến các cơ sở dữ liệu NoSQL thành lựa chọn tuyệt vời cho các trường hợp sử dụng cần thông lượng cao, độ trễ thấp cần thay đổi quy mô theo chiều ngang vượt qua giới hạn của một phiên bản duy nhất. |
| Hiệu năng | Hiệu năng thường phụ thuộc vào hệ thống con của ổ đĩa. Thông thường, việc tối ưu hóa các truy vấn, chỉ mục và cấu trúc bảng bắt buộc phải được thực hiện để đạt mức hiệu năng tối đa. | Hiệu năng thường được xem là chức năng của kích cỡ cụm phần cứng ngầm, độ trễ mạng và ứng dụng đưa ra lệnh gọi. |
| Quy mô | Cơ sở dữ liệu quan hệ thường tăng quy mô bằng cách tăng năng lực điện toán của phần cứng hoặc tăng quy mô bằng cách thêm bản sao của khối lượng công việc chỉ đọc. | Cơ sở dữ liệu NoSQL thường có tính phân mảnh cao do các mẫu truy cập khóa–giá trị có khả năng tăng quy mô bằng cách sử dụng kiến trúc được phân phối để tăng thông lượng, đem đến hiệu năng ổn định với quy mô gần như không giới hạn. |
| API | Yêu cầu lưu trữ và truy xuất dữ liệu được truyền đạt bằng cách sử dụng các truy vấn nhất quán với ngôn ngữ truy vấn có cấu trúc (SQL). Các truy vấn này được phân tích và thực thi bởi cơ sở dữ liệu quan hệ. | API trên cơ sở đối tượng cho phép các nhà phát triển ứng dụng dễ dàng lưu trữ và truy xuất cấu trúc dữ liệu trong bộ nhớ. Khóa phân mảnh tìm kiếm các cặp khóa–giá trị, tập hợp cột hoặc văn bản có cấu trúc chưa hoàn chỉnh có chứa đối tượng và thuộc tính của ứng dụng được xếp theo chuỗi. |

II/ Tổng quan về DynamoDB

Amazon DynamoDB là cơ sở dữ liệu về khóa-giá trị và tài liệu, với hiệu năng hoạt động chỉ trong vài mili giây ở mọi quy mô. Đây là một cơ sở dữ liệu bền vững, nhiều bản chính, đa vùng, được quản lý toàn phần, có tích hợp tính năng bảo mật, sao lưu, khôi phục và lưu vào bộ nhớ đệm cho các ứng dụng trên quy mô Internet. DynamoDB có thể xử lý hơn 10 nghìn tỷ yêu cầu/ngày và hỗ trợ tối đa hơn 20 triệu yêu cầu/giây.

Nhiều doanh nghiệp phát triển nhanh nhất thế giới như Lyft, Airbnb, Redfin, Samsung, Toyota và Capital One dựa vào quy mô và hiệu năng của DynamoDB để hỗ trợ khối lượng công việc then chốt của họ.

Hàng trăm nghìn khách hàng AWS đã lựa chọn DynamoDB làm cơ sở dữ liệu về khóa-giá trị và tài liệu cho các ứng dụng di động, web, trò chơi, công nghệ quảng cáo, IoT và các ứng dụng khác cần quyền truy cập dữ liệu với độ trễ thấp ở mọi quy mô. Tạo một bảng mới cho ứng dụng của bạn và để DynamoDB đảm nhiệm phần còn lại.

1. Lợi ích

* **Hiệu năng trên quy mô lớn**

DynamoDB hỗ trợ một số ứng dụng có quy mô lớn nhất thế giới bằng cách luôn đáp ứng thời gian phản hồi chỉ trong vài mili giây ở mọi quy mô. Bạn có thể xây dựng các ứng dụng gần như không giới hạn về thông lượng và dung lượng lưu trữ. Các bảng toàn cầu của DynamoDB sao chép dữ liệu của bạn trên nhiều Khu vực AWS để cấp cho bạn quyền truy cập nhanh, cục bộ vào dữ liệu cho các ứng dụng được phân phối trên toàn cầu. Đối với các trường hợp sử dụng đòi hỏi khả năng truy cập nhanh hơn nữa với độ trễ micrô giây, DynamoDB Accelerator (DAX) cung cấp bộ nhớ đệm trong bộ nhớ được quản lý toàn phần.

* **Không cần phải quản lý máy chủ**

DynamoDB là dịch vụ serverless khi bạn không cần phải cung cấp, vá hoặc quản lý máy chủ, cũng như không cần phải cài đặt, duy trì hoặc vận hành phần mềm. DynamoDB tự động thay đổi quy mô các bảng để điều chỉnh dung lượng và duy trì hiệu năng. Tích hợp dung sai cao và độ sẵn sàng hoạt động, nhờ đó ứng dụng của bạn không cần phải thiết kế các tính năng này. DynamoDB cung cấp cả chế độ dung lượng được cung cấp và chế độ dung lượng theo yêu cầu để bạn có thể tối ưu hóa chi phí bằng cách chỉ định công suất theo khối lượng công việc hoặc chỉ chi trả cho các tài nguyên mà bạn sử dụng.

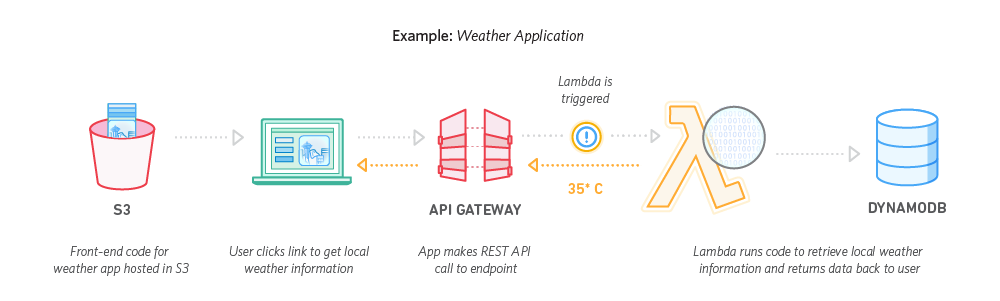
* **Phù hợp dùng trong doanh nghiệp**

DynamoDB hỗ trợ các giao tác ACID để bạn có thể xây dựng các ứng dụng quan trọng với doanh nghiệp trên quy mô lớn. DynamoDB mã hóa tất cả dữ liệu theo mặc định và cung cấp danh tính chi tiết cũng như quyền kiểm soát truy cập đối với mọi bảng của bạn. Bạn có thể tạo các bản sao lưu hoàn chỉnh của hàng trăm terabyte dữ liệu ngay tức thì mà không ảnh hưởng đến hiệu năng của bảng, đồng thời khôi phục kịp thời về điểm bất kỳ ở 35 ngày trước mà không có thời gian ngừng hoạt động. DynamoDB cũng được thỏa thuận mức dịch vụ hỗ trợ để đảm bảo khả năng hoạt động.

1. Ứng dụng

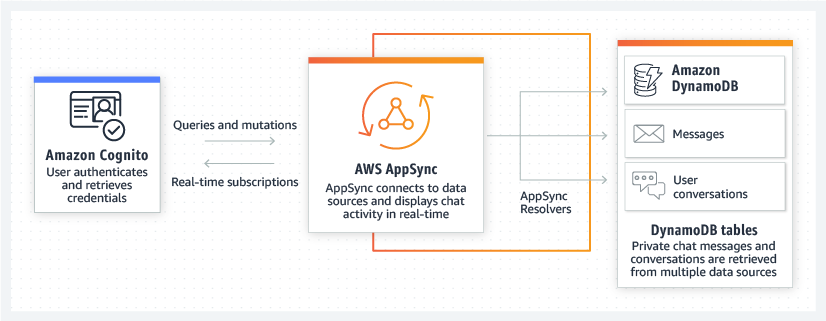
* **Ứng dụng web serverless**

Phát triển các ứng dụng web mạnh mẽ mà có thể tự động tăng và giảm quy mô. Bạn sẽ không cần phải duy trì những máy chủ và các ứng dụng của bạn sẽ tự động có độ khả dụng cao.



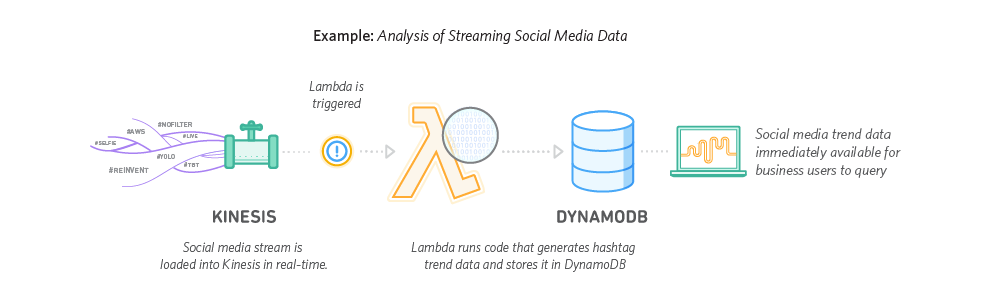
* **Backend di động**

Sử dụng DynamoDB và AWS AppSync để xây dựng các ứng dụng web và di động tương tác bằng các bản cập nhật theo thời gian thực, truy cập dữ liệu ngoại tuyến và đồng bộ hóa dữ liệu với tính năng xử lý xung đột tích hợp.



* **Vi dịch vụ**

Xây dựng các vi dịch vụ linh hoạt và có thể tái sử dụng nhờ dùng DynamoDB như một kho dữ liệu serverless để đạt hiệu năng nhanh và ổn định.



1. Trường hợp sử dụng

* **Công nghệ quảng cáo**

Các công ty trong ngành dọc công nghệ quảng cáo (CNQC) sử dụng DynamoDB làm kho lưu trữ khóa-giá trị để lưu trữ các loại dữ liệu tiếp thị khác nhau, chẳng hạn như hồ sơ người dùng, sự kiện người dùng, số lần nhấp chuột và liên kết được truy cập. Các trường hợp sử dụng hiện hành bao gồm cơ chế đấu giá quảng cáo trên thời gian thực (RTB), nhắm mục tiêu quảng cáo và phân bổ ngân sách tiếp thị. Các trường hợp sử dụng này đòi hỏi tốc độ yêu cầu cao (hàng triệu yêu cầu mỗi giây), độ tin cậy cao và độ trễ thấp có thể dự báo được. Các công ty sử dụng bộ đệm thông qua DynamoDB Accelerator (DAX) khi có khối lượng đọc cao hoặc cần độ trễ đọc chưa đến một mili giây. Các công ty công nghệ quảng cáo ngày càng cần triển khai RTB và nền tảng nhắm mục tiêu quảng cáo của họ ở nhiều Khu vực địa lý AWS, mà điều này đòi hỏi phải sao chép dữ liệu giữa các Khu vực.

**Trường hợp sử dụng thông dụng:**

* Kho hồ sơ người dùng trong RTB và nhắm mục tiêu quảng cáo
* Kho sự kiện người dùng, luồng dữ liệu nhấp chuột và dữ liệu lượt hiển thị
* Kho siêu dữ liệu cho tài sản
* Bộ đệm cho các mục thông dụng
* **Trò chơi**

Các công ty trong ngành dọc trò chơi sử dụng DynamoDB trong tất cả các tính năng của nền tảng trò chơi, bao gồm trạng thái trò chơi, dữ liệu người chơi, lịch sử phiên và bảng xếp hạng. Những lợi ích chính mà các công ty này nhận được từ DynamoDB là khả năng thay đổi quy mô đáng tin cậy cho hàng triệu người dùng và yêu cầu đồng thời, trong khi đảm bảo độ trễ thấp một cách ổn định dưới 10 mili-giây. Ngoài ra, là một dịch vụ được quản lý toàn phần, DynamoDB không phát sinh chi phí vận hành, vì vậy, các nhà phát triển trò chơi có thể tập trung vào phát triển trò chơi của họ thay vì quản lý cơ sở dữ liệu. Bên cạnh đó, khi các nhà phát triển trò chơi đang ngày càng tìm cách mở rộng từ một Khu vực AWS thành nhiều Khu vực AWS, họ có thể dựa vào các bảng toàn cầu của DynamoDB để sao chép dữ liệu đa Khu vực ở chế độ hiện hoạt-hiện hoạt.

**Trường hợp sử dụng thông dụng:**

* Trạng thái trò chơi
* Kho dữ liệu người chơi
* Kho dữ liệu lịch sử phiên của người chơi
* Bảng xếp hạng
* **Bán lẻ**

Nhiều công ty trong lĩnh vực bán lẻ sử dụng các mẫu thiết kế thông dụng của DynamoDB để cung cấp độ trễ thấp ổn định cho các trường hợp sử dụng tối quan trọng. Không còn phải bận tâm về thay đổi quy mô và gánh nặng vận hành là một lợi thế cạnh tranh quan trọng và là yếu tố thúc đẩy cho các sự kiện có tốc độ cao, quy mô cực lớn như Amazon Prime Day với cường độ rất khó dự báo. Khả năng tăng giảm quy mô cho phép những khách hàng này chỉ phải trả tiền cho công suất họ cần và giữ lại các nguồn lực kỹ thuật quý giá để tập trung vào đổi mới thay vì vận hành.

**Trường hợp sử dụng thông dụng:**

* Giỏ hàng
* Công cụ quy trình công việc
* Theo dõi và hoàn tất kiểm kê
* Hồ sơ và tài khoản khách hàng
* **Tài chính và ngân hàng**

Khi mà các công ty trong lĩnh vực tài chính ngân hàng xây dựng nhiều ứng dụng dành cho đám mây hơn, họ tìm cách sử dụng các dịch vụ được quản lý toàn phần để tăng cường tính linh hoạt, giảm bớt thời gian tung ra thị trường cũng như giảm thiểu chi phí vận hành. Đồng thời, họ phải đảm bảo độ bảo mật, độ tin cậy và độ sẵn sàng cao cho các ứng dụng của họ. Khi những công ty này mở rộng các dịch vụ hiện có được hỗ trợ bởi các hệ thống máy tính lớn, họ thấy rằng các hệ thống cũ không thể đáp ứng nhu cầu về khả năng mở rộng của cơ sở người dùng đang phát triển của họ, các nền tảng mới như ứng dụng di động và tăng lưu lượng truy cập. Để giải quyết vấn đề này, họ sao chép dữ liệu từ máy tính lớn của họ lên đám mây để giảm tải lưu lượng.

**Trường hợp sử dụng thông dụng:**

* Giao dịch người dùng
* Xử lý giao dịch theo sự kiện
* Phát hiện lừa đảo
* Giảm tải máy tính lớn và thu thập dữ liệu thay đổi
* **Truyền thông và giải trí**

Các công ty truyền thông và giải trí sử dụng DynamoDB khi họ yêu cầu một quy mô cực lớn về thông lượng và độ đồng thời, cũng như độ trễ thấp và độ tin cậy cao. DynamoDB thay đổi quy mô linh hoạt để xử lý tải và duy trì độ trễ thấp, điều rất quan trọng đối với các trường hợp theo thời gian thực, chẳng hạn như phát trực tuyến video và nội dung tương tác. Trong các trường hợp như vậy, số lượng người dùng đồng thời có thể lên tới hàng triệu người và không có cơ sở dữ liệu nào có thể xử lý mức độ đồng thời đó hiệu quả như DynamoDB. Mặc dù có tính đồng thời cao như vậy nhưng độ trễ vẫn đảm bảo thấp, mang lại trải nghiệm tối ưu cho người dùng cá nhân, cho dù là yêu cầu truy xuất phương tiện hay tham gia vào một sự kiện tương tác, theo thời gian thực. Những công ty này sử dụng DynamoDB để giải quyết các thách thức về khả năng thay đổi quy mô và tập trung vào phát triển tính năng thay vì quản lý cơ sở dữ liệu.

**Trường hợp sử dụng thông dụng:**

* Kho siêu dữ liệu phương tiện
* Kho dữ liệu người dùng
* Kho dữ liệu quản lý quyền kỹ thuật số
* **Phần mềm và internet**

Một điểm chung quan trọng giữa các công ty phần mềm và nhiều khách hàng khác của DynamoDB là quy mô internet. Trường hợp sử dụng của các công ty này yêu cầu khả năng đáp ứng mức độ đồng thời, tỷ lệ yêu cầu và lưu lượng truy cập tăng đột biến ở mức cực cao. Mức độ đồng thời này được đo lường theo hàng triệu người dùng và kết nối, trong khi tỷ lệ yêu cầu có thể dễ dàng đạt tới hàng triệu yêu cầu mỗi giây. DynamoDB đã được kiểm chứng là có khả năng xử lý các trường hợp sử dụng quy mô internet và các yêu cầu liên quan, đồng thời, duy trì độ trễ ổn định, chưa đến 10 mili-giây. Nhờ các bảng toàn cầu, khách hàng của DynamoDB có thể dễ dàng mở rộng ứng dụng của họ sang nhiều Khu vực AWS để có được tính liên tục trong hoạt động kinh doanh cùng khả năng tiếp cận toàn cầu.

**Trường hợp sử dụng thông dụng:**

* Kho siêu dữ liệu nội dung người dùng
* Kho dữ liệu biểu đồ mối quan hệ
* Bộ đệm siêu dữ liệu
* Kho dữ liệu theo dõi chuyến xe
* Kho dữ liệu người dùng, phương tiện và người lái
* Kho dữ liệu từ vựng người dùng

1. Các thương hiệu lớn sử dụng DynamoDB





* Nike Digital đã di chuyển các cụm Cassandra rộng của họ sang Amazon DynamoDB được quản lý toàn diện hơn, cho phép nhiều tài nguyên hơn để khách hàng có trải nghiệm tốt hơn.
* Oath đã triển khai tuân thủ GDPR đối với các ứng dụng của họ trên toàn cầu và tận dụng bảng toàn cầu của Amazon DynamoDB để hỗ trợ đồng bộ hóa dữ liệu.
* GE Aviation đã tái cấu trúc ứng dụng truy vấn dữ liệu và lập sơ đồ của họ để tiết kiệm chi phí, linh hoạt thay đổi quy mô và đạt hiệu năng cao bằng Amazon DynamoDB.
* Samsung Electronics sử dụng DynamoDB để sao lưu ứng dụng di động có kích thước petabyte, nhờ đó hiệu năng luôn ở mức cao và tiết kiệm chi phí.
* Và còn nhiều thương hiệu lớn khác nữa cũng tin dùng DynamoDB.

III/ Môi trường làm việc với DynamoDB

1. Cài đặt DynamoDB

Ngoài dịch vụ web Amazon DynamoDB, AWS cung cấp phiên bản có thể tải xuống của DynamoDB mà bạn có thể chạy trên máy tính của mình. Phiên bản có thể tải xuống cho phép bạn viết và kiểm tra các ứng dụng cục bộ mà không cần truy cập dịch vụ web của DynamoDB.

Ta có thể cài đặt DynamoDB thông qua 2 cách là: thiết lập DynamoDB trên máy tính (phiên bản có thể tải xuống) và dịch vụ web của DynamoDB.

**Thiết lập DynamoDB Local trên máy tính (Phiên bản có thể tải xuống)**

* B1: Truy cập vào link để tải phiên bản phù hợp với máy.

<https://docs.aws.amazon.com/en_us/amazondynamodb/latest/developerguide/DynamoDBLocal.DownloadingAndRunning.html>

* B2: Sau khi bạn tải xuống kho lưu trữ, giải nén nội dung và sao chép thư mục được trích xuất vào vị trí bạn chọn.
* B3: Để khởi động DynamoDB trên máy tính của bạn, hãy mở một cửa sổ nhắc lệnh, điều hướng đến thư mục mà bạn đã trích xuất DynamoDBLocal.jar và nhập lệnh sau.

**java -Djava.library.path=./DynamoDBLocal\_lib -jar DynamoDBLocal.jar -sharedDb**

* B4: Trước khi bạn có thể truy cập vào DynamoDB theo chương trình hoặc thông qua Giao diện dòng lệnh AWS (AWS CLI), bạn phải định cấu hình thông tin đăng nhập của mình để kích hoạt ủy quyền cho các ứng dụng của mình. DynamoDB có thể tải xuống yêu cầu bất kỳ thông tin đăng nhập nào để hoạt động, như trong ví dụ sau.

**AWS Access Key ID: "fakeMyKeyId"**

**AWS Secret Access Key: "fakeSecretAccessKey"**

* B5: Bạn có thể bắt đầu viết ứng dụng. Để truy cập vào DynamoDB chạy cục bộ, hãy sử dụng tham số --endpoint-url. Ví dụ: sử dụng lệnh sau để liệt kê các bảng DynamoDB.

**aws dynamodb list-tables --endpoint-url http://localhost:8000**

1. Kết nối vào DynamoDB

Bạn có thể truy cập Amazon DynamoDB bằng Console quản lý AWS, Giao diện dòng lệnh AWS (AWS CLI) hoặc API DynamoDB.

**Sử dụng Console**

Bạn có thể sử dụng Console để thực hiện các thao tác sau trong DynamoDB:

* Theo dõi các cảnh báo gần đây, tổng công suất, sức khỏe dịch vụ và tin tức mới nhất của DynamoDB trên bảng điều khiển của DynamoDB.
* Tạo, cập nhật và xóa bảng. Máy tính dung lượng cung cấp các ước tính về số lượng đơn vị năng lực yêu cầu dựa trên thông tin sử dụng bạn cung cấp.
* Quản lý luồng.
* Xem, thêm, cập nhật và xóa các mục được lưu trữ trong bảng. Quản lý thời gian để sống (TTL) để xác định khi các mục trong bảng hết hạn để có thể tự động xóa chúng khỏi cơ sở dữ liệu.
* Truy vấn và quét một bảng.
* Thiết lập và xem báo động để theo dõi việc sử dụng dung lượng bảng của bạn. Xem số liệu giám sát hàng đầu của bảng của bạn trên các biểu đồ thời gian thực từ CloudWatch.
* Sửa đổi công suất dự phòng của một bảng.
* Tạo và xóa các chỉ mục phụ toàn cầu.
* Tạo các kích hoạt để kết nối các luồng DynamoDB với các chức năng AWS Lambda.
* Áp dụng các thẻ cho tài nguyên của bạn để giúp tổ chức và xác định chúng.
* Mua dung lượng dự trữ.

Console hiển thị màn hình giới thiệu nhắc bạn tạo bảng đầu tiên. Để xem các bảng của bạn, từ ngăn điều hướng ở phía bên trái của Console, chọn Tables.

Dưới đây là tổng quan cấp cao về các hành động khả dụng trên mỗi bảng trong mỗi tab điều hướng:

* **Overview** – Xem chi tiết luồng và bảng và quản lý luồng và Thời gian để sống (TTL).
* **Items** – Quản lý các mục và thực hiện các truy vấn và quét.
* **Metrics** – Theo dõi số liệu CloudWatch.
* **Alarms** – Quản lý báo động CloudWatch.
* **Capacity** – Sửa đổi công suất dự phòng của một bảng.
* **Indexes** – Quản lý các chỉ số phụ toàn cầu.
* **Triggers** – Quản lý các kích hoạt để kết nối các luồng DynamoDB với các chức năng Lambda.
* **Access control** – Thiết lập kiểm soát truy cập chi tiết với liên kết nhận dạng web.
* **Tags** – Áp dụng các thẻ cho tài nguyên của bạn để giúp tổ chức và xác định chúng.

**Sử dụng AWS CLI:**

Định dạng dòng lệnh bao gồm tên hoạt động của DynamoDB theo sau là các tham số cho hoạt động đó. AWS CLI hỗ trợ cú pháp nhanh cho các giá trị tham số, cũng như JSON.

Ví dụ: lệnh sau tạo một bảng có tên là Âm nhạc. Khóa phân vùng là Artist và khóa sắp xếp là SongTitle. (Để dễ đọc hơn, các lệnh dài trong phần này được chia thành các dòng riêng biệt.)

aws dynamodb create-table \

--table-name Music \

--attribute-definitions \

AttributeName=Artist,AttributeType=S \

AttributeName=SongTitle,AttributeType=S \

--key-schema AttributeName=Artist,KeyType=HASH AttributeName=SongTitle,KeyType=RANGE \

--provisioned-throughput ReadCapacityUnits=1,WriteCapacityUnits=1

Các lệnh sau thêm các mục mới vào bảng. Các ví dụ này sử dụng kết hợp cú pháp nhanh và JSON

aws dynamodb put-item \

--table-name Music \

--item \

'{"Artist": {"S": "No One You Know"}, "SongTitle": {"S": "Call Me Today"}, "AlbumTitle": {"S": "Somewhat Famous"}}' \

--return-consumed-capacity TOTAL

aws dynamodb put-item \

--table-name Music \

--item '{ \

"Artist": {"S": "Acme Band"}, \

"SongTitle": {"S": "Happy Day"}, \

"AlbumTitle": {"S": "Songs About Life"} }' \

--return-consumed-capacity TOTAL

**Với phiên bản DynamoDB Local:**

AWS CLI cũng có thể tương tác với DynamoDB (Phiên bản có thể tải xuống) chạy trên máy tính của bạn. Để kích hoạt tính năng này, hãy thêm tham số sau vào mỗi lệnh:

--endpoint-url http: // localhost: 8000

Ví dụ sau sử dụng AWS CLI để liệt kê các bảng trong cơ sở dữ liệu cục bộ.

aws dynamodb list-tables --endpoint-url http://localhost:8000

Nếu DynamoDB đang sử dụng số cổng khác với mặc định (8000), hãy sửa đổi giá trị --endpoint-url cho phù hợp.

***Lưu ý:*** AWS CLI không thể sử dụng phiên bản có thể tải xuống của DynamoDB làm điểm cuối mặc định. Do đó, bạn phải chỉ định --endpoint-url với mỗi lệnh.

Tham khảo ở link dưới để biết thêm chi tiết:

<https://docs.aws.amazon.com/en_us/amazondynamodb/latest/developerguide/Tools.CLI.html>

**Sử dụng API**

Bạn có thể sử dụng AWS Management Console và AWS Command Line để hoạt động tương tác với Amazon DynamoDB. Tuy nhiên, để tận dụng tối đa DynamoDB, bạn có thể viết mã ứng dụng bằng SDK AWS.

SDK AWS cung cấp hỗ trợ rộng rãi cho DynamoDB bằng Java, JavaScript trong trình duyệt, .NET, Node.js, PHP, Python, Ruby, C ++, Go, Android và iOS.

Trước khi bạn có thể sử dụng SDK AWS với DynamoDB, bạn phải có ID khóa truy cập AWS và khóa truy cập bí mật.

Tham khảo ở link dưới để biết thêm chi tiết:

<https://docs.aws.amazon.com/en_us/amazondynamodb/latest/developerguide/Using.API.html>

IV/ Tìm hiểu về DynamoDB

1. Các Components trong DynamoDB

Trong DynamoDB, các ***tables*** (bảng), ***items*** (mục) và ***attributes*** (thuộc tính) là các thành phần cốt lõi để bạn làm việc. ***Bảng*** là một tập hợp các ***mục*** và mỗi ***mục*** là một tập hợp các ***thuộc tính***. DynamoDB sử dụng các ***khóa chính*** (Primary key) để xác định duy nhất từng ***mục*** trong ***bảng*** và các ***chỉ mục phụ (secondary indexes)*** để cung cấp thêm tính linh hoạt truy vấn. Bạn có thể sử dụng DynamoDB Streams để ghi lại các sự kiện sửa đổi dữ liệu trong các bảng DynamoDB.

* **Tables** – Tương tự như các hệ thống cơ sở dữ liệu khác, DynamoDB lưu trữ dữ liệu trong các bảng. Một bảng là một tập hợp dữ liệu.
* **Items** – Mỗi bảng chứa 0 hoặc nhiều mục. Một mục là một nhóm các thuộc tính có thể nhận dạng duy nhất trong số tất cả các mục khác. Các mục trong DynamoDB tương tự như các rows (hàng), record (bản ghi) hoặc tuples (bộ dữ liệu) trong các hệ thống cơ sở dữ liệu khác. Trong DynamoDB, không có giới hạn về số lượng vật phẩm bạn có thể lưu trữ trong một bảng.
* **Attributes** – Mỗi mục bao gồm một hoặc nhiều thuộc tính. Một thuộc tính là một yếu tố dữ liệu cơ bản. Các thuộc tính trong DynamoDB tương tự như các trường hoặc cột trong các hệ thống cơ sở dữ liệu khác.

**Ví dụ:** PEOPLE

{

“PersonID”: 101,

“LastName”: “Smith”,

“FirstName”: “Fred”,

“Phone”: “555-4321”

}

Lưu ý về bảng People:

{

“PersonID”: 103,

“LastName”: “Steve”,

“FirstName”: “Paul”,

“Address”: {

“Street”: “123 Main”,

“City”: “London”,

}

“FarvoriteColor”: “Blue”

}

{

“PersonID”: 102,

“LastName”: “Jones”,

“FirstName”: “Mary”,

“Address”: {

“Street”: “123 Main”,

“City”: “Anytown”,

“State”: “OH”,

}

}

* Mỗi mục trong bảng có một mã định danh duy nhất hoặc khóa chính để phân biệt mục đó với tất cả các mục khác trong bảng. Trong bảng People, khóa chính bao gồm một thuộc tính (PersonID).
* Khác với khóa chính, bảng People là schemaless, có nghĩa là cả hai thuộc tính và kiểu dữ liệu của chúng đều không cần được xác định trước. Mỗi mục có thể có các thuộc tính riêng biệt của nó.
* Hầu hết các thuộc tính là vô hướng, có nghĩa là chúng chỉ có thể có một giá trị. Chuỗi và số là ví dụ phổ biến của vô hướng.
* Một số mục có thuộc tính lồng nhau (Địa chỉ). DynamoDB hỗ trợ các thuộc tính lồng nhau sâu tới 32 cấp.

**Primary Key**

Khi bạn tạo bảng, ngoài tên bảng, bạn phải chỉ định khóa chính của bảng. Khóa chính xác định duy nhất từng mục trong bảng, do đó không có hai mục nào có thể có cùng một khóa.

DynamoDB hỗ trợ hai loại khóa chính khác nhau:

* **Partition key** (Khóa phân vùng) - Khóa chính đơn giản, bao gồm một thuộc tính được gọi là khóa phân vùng. DynamoDB sử dụng giá trị của khóa phân vùng làm đầu vào cho hàm băm bên trong. Đầu ra từ hàm băm xác định phân vùng (lưu trữ vật lý bên trong cho DynamoDB) trong đó mục sẽ được lưu trữ. Trong một bảng chỉ có khóa phân vùng, không có hai mục nào có thể có cùng giá trị khóa phân vùng.
* **Partition key and sort key (**Khóa phân vùng và khóa sắp xếp) - Được gọi là khóa chính tổng hợp, loại khóa này bao gồm hai thuộc tính. Thuộc tính đầu tiên là khóa phân vùng và thuộc tính thứ hai là khóa sắp xếp. DynamoDB sử dụng giá trị khóa phân vùng làm đầu vào cho hàm băm bên trong. Đầu ra từ hàm băm xác định phân vùng (lưu trữ vật lý bên trong cho DynamoDB) trong đó mục sẽ được lưu trữ. Tất cả các mục có cùng giá trị khóa phân vùng được lưu trữ cùng nhau, theo thứ tự được sắp xếp theo giá trị khóa sắp xếp. Trong một bảng có khóa phân vùng và khóa sắp xếp, hai mục có thể có cùng giá trị khóa phân vùng. Tuy nhiên, hai mục đó phải có các giá trị khóa sắp xếp khác nhau. Khóa chính tổng hợp cung cấp cho bạn sự linh hoạt bổ sung khi truy vấn dữ liệu.

***Lưu ý:***

* *Khóa phân vùng của một mục còn được gọi là thuộc tính băm của nó. Thuật ngữ băm thuộc tính xuất phát từ việc sử dụng hàm băm nội bộ trong DynamoDB, phân phối đồng đều các mục dữ liệu trên các phân vùng, dựa trên các giá trị khóa phân vùng của chúng.*
* *Khóa sắp xếp của một mục còn được gọi là thuộc tính phạm vi của nó. Thuộc tính phạm vi thuật ngữ xuất phát từ cách DynamoDB lưu trữ các mục có cùng khóa phân vùng gần nhau về mặt vật lý, theo thứ tự được sắp xếp theo giá trị khóa sắp xếp.*

Mỗi thuộc tính khóa chính phải là vô hướng (có nghĩa là nó chỉ có thể chứa một giá trị duy nhất). Các kiểu dữ liệu duy nhất được phép cho các thuộc tính khóa chính là chuỗi, số hoặc nhị phân. Không có hạn chế như vậy đối với các thuộc tính khác, không chính.

**Secondary Indexes**

Bạn có thể tạo một hoặc nhiều chỉ mục phụ trên bảng. Một chỉ mục phụ cho phép bạn truy vấn dữ liệu trong bảng bằng khóa thay thế, ngoài các truy vấn đối với khóa chính. DynamoDB không yêu cầu bạn sử dụng các chỉ mục, nhưng chúng giúp ứng dụng của bạn linh hoạt hơn khi truy vấn dữ liệu của bạn. Sau khi bạn tạo một chỉ mục phụ trên bảng, bạn có thể đọc dữ liệu từ chỉ mục theo cách tương tự như bạn làm từ bảng.

DynamoDB hỗ trợ hai loại chỉ mục:

* Global secondary index (Chỉ mục phụ toàn cầu) - Một chỉ mục có khóa phân vùng và khóa sắp xếp có thể khác với chỉ mục trên bảng.
* Local secondary index (Chỉ mục phụ cục bộ) - Một chỉ mục có cùng khóa phân vùng với bảng, nhưng một khóa sắp xếp khác.

Mỗi bảng trong DynamoDB có giới hạn 20 chỉ mục phụ toàn cầu (giới hạn mặc định) và 5 chỉ mục phụ cục bộ trên mỗi bảng.

**DynamoDB Streams**

DynamoDB Streams là một tính năng tùy chọn để ghi lại các sự kiện sửa đổi dữ liệu trong các bảng của DynamoDB. Dữ liệu về các sự kiện này xuất hiện trong luồng trong thời gian gần như thực và theo thứ tự các sự kiện đã xảy ra.

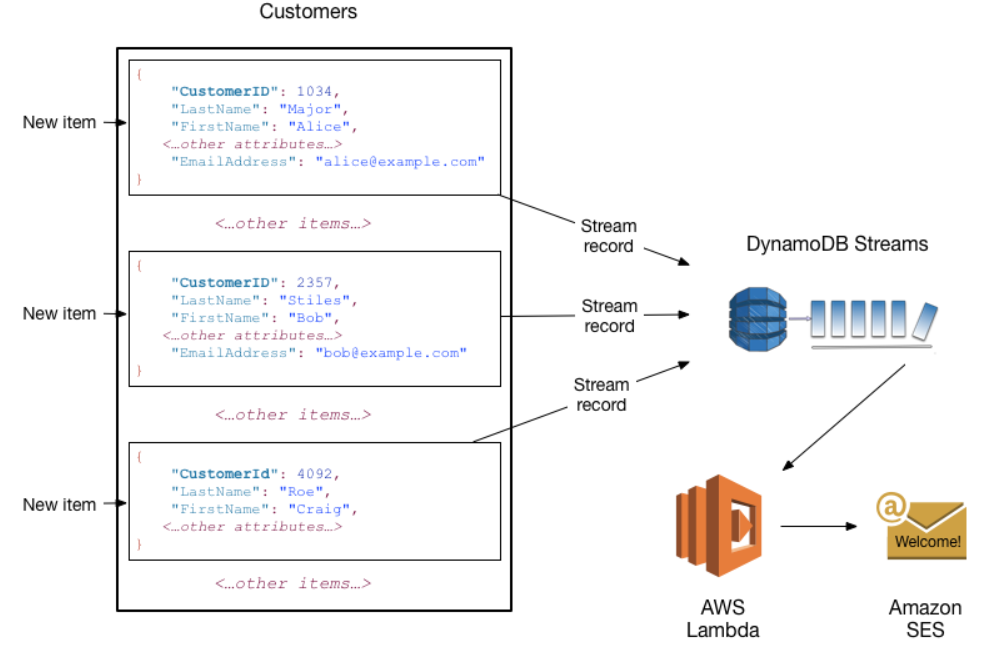
Mỗi sự kiện được đại diện bởi một bản ghi dòng. Nếu bạn bật luồng trên bảng, Luồng Động sẽ ghi bản ghi luồng bất cứ khi nào một trong các sự kiện sau xảy ra:

* Một mục mới được thêm vào bảng: Luồng chụp một hình ảnh của toàn bộ mục, bao gồm tất cả các thuộc tính của nó.
* Một mục được cập nhật: Luồng chụp ảnh "trước" và "sau" của bất kỳ thuộc tính nào đã được sửa đổi trong mục.
* Một mục bị xóa khỏi bảng: Luồng chụp một hình ảnh của toàn bộ mục trước khi nó bị xóa.

Mỗi bản ghi luồng cũng chứa tên của bảng, dấu thời gian sự kiện và siêu dữ liệu khác. Luồng hồ sơ có thời gian tồn tại là 24 giờ; sau đó, chúng sẽ tự động bị xóa khỏi luồng.

Bạn có thể sử dụng DynamoDB Streams cùng với AWS Lambda để tạo mã kích hoạt, mã tự động thực thi bất cứ khi nào một sự kiện đáng quan tâm xuất hiện trong luồng.

**Ví dụ:** Xem xét bảng Khách hàng có chứa thông tin khách hàng cho một công ty. Giả sử bạn muốn gửi email "chào mừng" tới từng khách hàng mới. Bạn có thể kích hoạt một luồng trên bảng đó và sau đó liên kết luồng với chức năng Lambda. Hàm Lambda sẽ thực thi bất cứ khi nào một bản ghi luồng mới xuất hiện, nhưng chỉ xử lý các mục mới được thêm vào bảng Khách hàng. Đối với bất kỳ mục nào có thuộc tính EmailAddress, chức năng Lambda sẽ gọi Dịch vụ email đơn giản của Amazon (Amazon SES) để gửi email đến địa chỉ đó.



Ngoài các trình kích hoạt, DynamoDB Streams cho phép các giải pháp mạnh mẽ như sao chép dữ liệu trong và trên các Vùng AWS, các chế độ xem được cụ thể hóa của dữ liệu trong các bảng của DynamoDB, phân tích dữ liệu bằng cách sử dụng các khung nhìn cụ thể của Kinesis và hơn thế nữa.

1. DynamoDB API

Để làm việc với Amazon DynamoDB, ứng dụng của bạn phải sử dụng một vài thao tác API đơn giản. Sau đây là tóm tắt về các hoạt động này, được tổ chức theo thể loại.

**Control Plane**

*Control Plane* cho phép bạn tạo và quản lý các bảng DynamoDB. Chúng cũng cho phép bạn làm việc với các chỉ mục, luồng và các đối tượng khác phụ thuộc vào bảng.

* CreateTable – Tạo một bảng mới. Theo tùy chọn, bạn có thể tạo một hoặc nhiều chỉ mục phụ và bật DynamoDB Streams cho bảng.
* DescribeTable – Trả về thông tin về một bảng, chẳng hạn như lược đồ khóa chính, cài đặt thông lượng và thông tin chỉ mục.
* ListTables – Trả về tên của tất cả các bảng của bạn trong một danh sách.
* UpdateTable – Sửa đổi cài đặt của bảng hoặc chỉ mục của bảng, tạo hoặc xóa chỉ mục mới trên bảng hoặc sửa đổi cài đặt DynamoDB cho bảng.
* DeleteTable – Xóa một bảng và tất cả các đối tượng phụ thuộc của nó khỏi DynamoDB.

**Data Plane**

*Data Plane* cho phép bạn thực hiện các hành động tạo, đọc, cập nhật và xóa (còn gọi là CRUD) trên dữ liệu trong một bảng. Một số hoạt động của mặt phẳng dữ liệu cũng cho phép bạn đọc dữ liệu từ một chỉ mục phụ.

* Tạo dữ liệu
  + PutItem – Viết một mục duy nhất vào một bảng. Bạn phải chỉ định các thuộc tính khóa chính, nhưng bạn không phải chỉ định các thuộc tính khác.
  + BatchWriteItem – Viết tối đa 25 mục vào một bảng. Điều này hiệu quả hơn so với việc gọi PutItem nhiều lần vì ứng dụng của bạn chỉ cần một chuyến đi khứ hồi mạng duy nhất để viết các mục. Bạn cũng có thể sử dụng BatchWriteItem để xóa nhiều mục từ một hoặc nhiều bảng.
* Đọc dữ liệu
  + GetItem – Lấy một mục duy nhất từ một bảng. Bạn phải chỉ định khóa chính cho mục mà bạn muốn. Bạn có thể truy xuất toàn bộ mục hoặc chỉ là một tập hợp con các thuộc tính của nó.
  + BatchGetItem – Lấy tối đa 100 mục từ một hoặc nhiều bảng. Điều này hiệu quả hơn so với việc gọi GetItem nhiều lần vì ứng dụng của bạn chỉ cần một chuyến đi khứ hồi mạng duy nhất để đọc các mục.
  + Query – Lấy tất cả các mục có một khóa phân vùng cụ thể. Bạn phải chỉ định giá trị khóa phân vùng. Bạn có thể truy xuất toàn bộ các mục hoặc chỉ là một tập hợp con các thuộc tính của chúng. Theo tùy chọn, bạn có thể áp dụng một điều kiện cho các giá trị khóa sắp xếp để bạn chỉ truy xuất một tập hợp con của dữ liệu có cùng khóa phân vùng. Bạn có thể sử dụng thao tác này trên một bảng, với điều kiện bảng có cả khóa phân vùng và khóa sắp xếp. Bạn cũng có thể sử dụng thao tác này trên một chỉ mục, với điều kiện là chỉ mục đó có cả khóa phân vùng và khóa sắp xếp.
  + Scan – Lấy tất cả các mục trong bảng hoặc chỉ mục được chỉ định. Bạn có thể truy xuất toàn bộ các mục hoặc chỉ là một tập hợp con các thuộc tính của chúng. Tùy chọn, bạn có thể áp dụng điều kiện lọc để chỉ trả về các giá trị mà bạn quan tâm và loại bỏ phần còn lại.
* Cập nhật dữ liệu
  + UpdateItem – Sửa đổi một hoặc nhiều thuộc tính trong một mục. Bạn phải chỉ định khóa chính cho mục bạn muốn sửa đổi. Bạn có thể thêm các thuộc tính mới và sửa đổi hoặc loại bỏ các thuộc tính hiện có. Bạn cũng có thể thực hiện các cập nhật có điều kiện, để cập nhật chỉ thành công khi đáp ứng điều kiện do người dùng xác định. Tùy chọn, bạn có thể thực hiện một bộ đếm nguyên tử, tăng hoặc giảm một thuộc tính số mà không can thiệp vào các yêu cầu ghi khác.
* Xóa dữ liệu
  + DeleteItem – Xóa một mục duy nhất từ một bảng. Bạn phải chỉ định khóa chính cho mục bạn muốn xóa.
  + BatchWriteItem – Xóa tối đa 25 mục từ một hoặc nhiều bảng. Điều này hiệu quả hơn so với việc gọi DeleteItem nhiều lần vì ứng dụng của bạn chỉ cần một chuyến đi khứ hồi mạng duy nhất để xóa các mục. Bạn cũng có thể sử dụng BatchWriteItem để thêm nhiều mục vào một hoặc nhiều bảng.

**DynamoDB Streams**

*DynamoDB Streams* cho phép bạn bật hoặc tắt luồng trên bảng và cho phép truy cập vào các bản ghi sửa đổi dữ liệu có trong luồng.

* ListStreams – Trả về danh sách tất cả các luồng của bạn hoặc chỉ luồng cho một bảng cụ thể.
* DescribeStream – Trả về thông tin về một luồng, chẳng hạn như Tên tài nguyên Amazon (ARN) và nơi ứng dụng của bạn có thể bắt đầu đọc một vài bản ghi luồng đầu tiên.
* GetShardIterator – Trả về một trình lặp shard, đây là cấu trúc dữ liệu mà ứng dụng của bạn sử dụng để truy xuất các bản ghi từ luồng.
* GetRecords – Lấy một hoặc nhiều bản ghi luồng, sử dụng trình lặp shard cho trước.

**Transactions**

*Transactions* cung cấp tính nguyên tử, tính nhất quán, sự cô lập và độ bền (ACID) cho phép bạn duy trì tính chính xác của dữ liệu trong các ứng dụng của mình dễ dàng hơn.

* TransactWriteItems – Một hoạt động hàng loạt cho phép các hoạt động Đặt, Cập nhật và Xóa thành nhiều mục cả trong và trên các bảng với kết quả được bảo đảm tất cả hoặc không có gì.
* TransactGetItems – Một hoạt động hàng loạt cho phép các hoạt động Nhận lấy nhiều mục từ một hoặc nhiều bảng.

1. Cách đặt tên và kiểu dữ liệu

**a. Cách đặt tên:**

Các bảng, thuộc tính và các đối tượng khác trong DynamoDB phải có tên. Tên nên có ý nghĩa và ngắn gọn.

Sau đây là các quy tắc đặt tên cho DynamoDB:

* Tất cả các tên phải được mã hóa bằng UTF-8 và phân biệt chữ hoa chữ thường.
* Tên bảng và tên chỉ mục phải dài từ 3 đến 255 ký tự và chỉ có thể chứa các ký tự sau:
  + a-z
  + A-Z
  + 0-9
  + \_ (gạch dưới)
  + - (gạch ngang)
  + . (chấm)
* Tên thuộc tính phải dài từ 1 đến 255 ký tự.

**Từ dành riêng và ký tự đặc biệt**

DynamoDB có một danh sách các từ dành riêng và các ký tự đặc biệt. Ngoài ra, các ký tự sau có ý nghĩa đặc biệt trong DynamoDB: # (hash) và : (colon).

* Trong biểu thức điều kiện, : (ký tự dấu hai chấm) biểu thị giá trị thuộc tính biểu thức, giữ chỗ cho một giá trị thực

Mặc dù DynamoDB cho phép bạn sử dụng các từ dành riêng và các ký tự đặc biệt này cho tên, chúng tôi khuyên bạn nên tránh làm như vậy vì bạn phải xác định các biến giữ chỗ bất cứ khi nào bạn sử dụng các tên này trong một biểu thức. Để biết thêm thông tin, hãy xem Tên thuộc tính biểu thức.

**b. Kiểu dữ liệu**

DynamoDB hỗ trợ nhiều loại dữ liệu khác nhau cho các thuộc tính trong một bảng. Chúng có thể được phân loại như sau:

* **Scalar Types** – Một loại vô hướng có thể đại diện chính xác một giá trị. Các kiểu vô hướng là Number, String, Binary, Boolean và NULL.
* **Document Types** – Một loại tài liệu có thể biểu diễn một cấu trúc phức tạp với các thuộc tính lồng nhau, như bạn sẽ tìm thấy trong một tài liệu JSON. Các loại tài liệu là danh sách và bản đồ.
* **Set Types** – Một kiểu thiết lập có thể đại diện cho nhiều giá trị vô hướng. Các loại tập hợp là tập hợp chuỗi, tập số và tập nhị phân.

Khi bạn tạo bảng hoặc chỉ mục phụ, bạn phải chỉ định tên và loại dữ liệu của từng thuộc tính khóa chính (khóa phân vùng và khóa sắp xếp). Hơn nữa, mỗi thuộc tính khóa chính phải được xác định là chuỗi kiểu, số hoặc nhị phân.

DynamoDB là một cơ sở dữ liệu NoQuery và là schemaless. Điều này có nghĩa là, ngoài các thuộc tính khóa chính, bạn không phải xác định bất kỳ thuộc tính hoặc loại dữ liệu nào khi bạn tạo bảng. Khi so sánh, cơ sở dữ liệu quan hệ yêu cầu bạn xác định tên và loại dữ liệu của từng cột khi bạn tạo bảng.

**Scalar Types:** Gồm kiểu: Number, String, Binary, Boolean và NULL.

**Number (Số)**

Các số có thể là dương, âm hoặc bằng không. Các số có thể có tới 38 chữ số chính xác. Vượt quá kết quả này trong một ngoại lệ.

* Phạm vi tích cực: 1E-130 đến 9.9999999999999999999999999999999999999E+125
* Phạm vi tiêu cực: -9.9999999999999999999999999999999999999E+125 to -1E-130

Trong DynamoDB, các số được biểu diễn dưới dạng độ dài thay đổi. Các số 0 đứng đầu và dấu được cắt.

Tất cả các số được gửi qua mạng tới DynamoDB dưới dạng chuỗi, để tối đa hóa khả năng tương thích giữa các ngôn ngữ và thư viện. Tuy nhiên, DynamoDB coi chúng là các thuộc tính loại số cho các hoạt động toán học.

***Lưu ý:*** Nếu độ chính xác của số là quan trọng, bạn nên chuyển số cho DynamoDB bằng cách sử dụng các chuỗi mà bạn chuyển đổi từ loại số.

**String (Chuỗi)**

Chuỗi là Unicode với mã hóa nhị phân UTF-8. Độ dài của chuỗi phải lớn hơn 0 và bị ràng buộc bởi giới hạn kích thước mục tối đa của DynamoDB là **400 KB.**

Các ràng buộc bổ sung sau áp dụng cho các thuộc tính khóa chính được xác định là chuỗi kiểu:

* Đối với khóa chính đơn giản, độ dài tối đa của giá trị thuộc tính đầu tiên (khóa phân vùng) là 2048 byte.
* Đối với khóa chính tổng hợp, độ dài tối đa của giá trị thuộc tính thứ hai (khóa sắp xếp) là 1024 byte.

DynamoDB đối chiếu và so sánh các chuỗi bằng cách sử dụng các byte của mã hóa chuỗi UTF-8 bên dưới. Ví dụ: "a" (0x61) lớn hơn "A" (0x41) và "" (0xC2BF) lớn hơn "z" (0x7A).

**Binary (Chuỗi nhị phân)**

Thuộc tính loại nhị phân có thể lưu trữ bất kỳ dữ liệu nhị phân nào, chẳng hạn như văn bản nén, dữ liệu được mã hóa hoặc hình ảnh. Bất cứ khi nào DynamoDB so sánh các giá trị nhị phân, nó sẽ coi mỗi byte của dữ liệu nhị phân là không dấu.

Độ dài của thuộc tính nhị phân phải lớn hơn 0 và bị giới hạn bởi giới hạn kích thước mục của DynamoDB tối đa là **400 KB.**

Nếu bạn xác định thuộc tính khóa chính là thuộc tính loại nhị phân, các ràng buộc bổ sung sau sẽ được áp dụng:

* Đối với khóa chính đơn giản, độ dài tối đa của giá trị thuộc tính đầu tiên (khóa phân vùng) là 2048 byte.
* Đối với khóa chính tổng hợp, độ dài tối đa của giá trị thuộc tính thứ hai (khóa sắp xếp) là 1024 byte.

Các ứng dụng của bạn phải mã hóa các giá trị nhị phân ở định dạng được mã hóa base64 trước khi gửi chúng đến DynamoDB. Khi nhận được các giá trị này, DynamoDB sẽ giải mã dữ liệu thành một mảng byte không dấu và sử dụng nó làm độ dài của thuộc tính nhị phân.

**Boolean (Luận lý):** Thuộc tính loại Boolean có thể lưu trữ True (đúng) hoặc False (sai).

**NULL:** Null đại diện cho một thuộc tính với trạng thái không xác định hoặc không xác định.

**Document Types**

Bao gồm kiểu là loại List (danh sách) và Map (bản đồ). Các kiểu dữ liệu này có thể được lồng vào nhau, để biểu diễn các cấu trúc dữ liệu phức tạp sâu tới 32 cấp độ.

Không có giới hạn về số lượng giá trị trong danh sách hoặc bản đồ, miễn là mục chứa giá trị phù hợp với giới hạn kích thước mục của DynamoDB (400 KB).

Giá trị thuộc tính không thể là chuỗi rỗng hoặc tập rỗng (tập chuỗi, tập số hoặc tập nhị phân). Tuy nhiên, danh sách trống và bản đồ được cho phép.

**List (danh sách)**

Một thuộc tính loại danh sách có thể lưu trữ một bộ sưu tập các giá trị. Danh sách được đặt trong dấu ngoặc vuông: [...]

Một danh sách tương tự như một mảng JSON. Không có hạn chế về các loại dữ liệu có thể được lưu trữ trong một phần tử danh sách và các phần tử trong một phần tử danh sách không phải cùng loại.

Ví dụ sau đây cho thấy một danh sách chứa hai chuỗi và một số.

FavoriteThings: ["Cookies", "Coffee", 3.14159]

***Lưu ý:*** DynamoDB cho phép bạn làm việc với các phần tử riêng lẻ trong danh sách, ngay cả khi các phần tử đó được lồng sâu

**Map (bản đồ)**

Thuộc tính loại bản đồ có thể lưu trữ một tập hợp các cặp tên-giá trị không có thứ tự. Các bản đồ được đặt trong các dấu ngoặc nhọn: {...}

Một bản đồ tương tự như một đối tượng JSON. Không có hạn chế về các loại dữ liệu có thể được lưu trữ trong một phần tử bản đồ và các phần tử trong bản đồ không phải cùng loại.

Bản đồ là lý tưởng để lưu trữ các tài liệu JSON trong DynamoDB. Ví dụ sau đây cho thấy một bản đồ có chứa một chuỗi, một số và một danh sách lồng nhau có chứa một bản đồ khác.

{

Day: "Monday",

UnreadEmails: 42,

ItemsOnMyDesk: [

"Coffee Cup",

"Telephone",

{

Pens: { Quantity : 3},

Pencils: { Quantity : 2},

Erasers: { Quantity : 1}

}

]

}

***Lưu ý:*** DynamoDB cho phép bạn làm việc với các yếu tố riêng lẻ trong bản đồ, ngay cả khi các yếu tố đó được lồng sâu

**Set Types**

DynamoDB hỗ trợ các loại đại diện cho các bộ giá trị số, chuỗi hoặc nhị phân. Tất cả các yếu tố trong một bộ phải cùng loại.

Không có giới hạn về số lượng giá trị trong một bộ, miễn là mục chứa giá trị phù hợp với giới hạn kích thước mục của DynamoDB (400 KB).

Mỗi giá trị trong một bộ phải là duy nhất. Thứ tự của các giá trị trong một tập hợp không được bảo toàn. Do đó, các ứng dụng của bạn không được dựa vào bất kỳ thứ tự cụ thể nào của các thành phần trong tập hợp. Cuối cùng, DynamoDB không hỗ trợ các tập hợp trống.

Ví dụ sau đây cho thấy một bộ chuỗi, một bộ số và một bộ nhị phân:

["Black", "Green", "Red"]

[42.2, -19, 7.5, 3.14]

["U3Vubnk=", "UmFpbnk=", "U25vd3k="]

V/ Lập trình DynamoDB với ngôn ngữ JavaScript

AWS SDK dành cho JavaScript cho phép bạn truy cập trực tiếp các dịch vụ AWS từ mã JavaScript chạy trên trình duyệt. Xác thực người dùng thông qua Facebook, Google hoặc Đăng nhập vào Amazon bằng liên kết danh tính trên web. Lưu trữ dữ liệu ứng dụng trên Amazon DynamoDB và lưu hồ sơ người dùng lên Amazon S3. Bạn chỉ cần một thẻ tập lệnh để bắt đầu sử dụng SDK.

1. Tính năng

* **Đăng nhập bằng mạng xã hội**

Bảo mật việc truy cập của ứng dụng web tới các tài nguyên AWS mà không cần bất kỳ mã phía máy chủ nào. AWS SDK dành cho JavaScript giúp bạn dễ dàng tích hợp với [Amazon Cognito](https://aws.amazon.com/cognito/), việc này cho phép bạn cung cấp cho người dùng đã xác thực và chưa xác thực các mức truy cập API tùy chỉnh linh hoạt. Dễ dàng thiết lập Amazon Cognito bằng Facebook, Google, Login with Amazon và bất kỳ nhà cung cấp danh tính OpenID Connect nào.

* **Dữ liệu trực tuyến**

Mang lại trải nghiệm ứng dụng trên trình duyệt có kết nối bằng cách sử dụng dữ liệu trên đám mây. [Amazon DynamoDB](https://aws.amazon.com/dynamodb/) là dịch vụ cơ sở dữ liệu NoSQL được quản lý toàn phần, có tốc độ truy cập nhanh giúp cho việc lưu trữ và truy xuất dữ liệu ứng dụng của bạn trở nên đơn giản và có mức chi phí hợp lý. [Kiểm soát truy cập phân định tinh tế](http://docs.aws.amazon.com/amazondynamodb/latest/developerguide/FGAC_DDB.html) cho phép cách ly dữ liệu của người dùng cụ thể một cách bảo mật và cho phép lưu trữ dữ liệu cá nhân cho từng người dùng của bạn trực tiếp từ một ứng dụng trên trình duyệt.

* **Lưu trữ tập tin**

Lưu trữ và truy cập số lượng không giới hạn tập tin người dùng một cách bảo mật bằng cách sử dụng [Amazon S3](https://aws.amazon.com/s3/). AWS SDK dành cho JavaScript tích hợp với giao diện tập tin riêng của trình duyệt để đơn giản hóa quá trình tải lên Amazon S3.

* **Hỗ trợ trình duyệt**

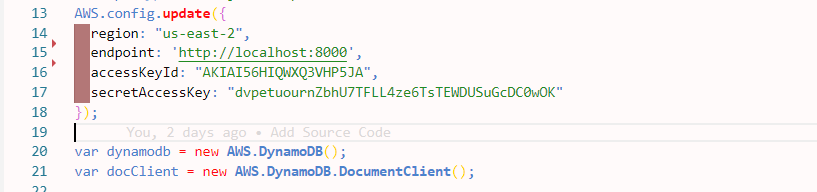
AWS SDK dành cho JavaScript trên trình duyệt được hỗ trợ bởi tất cả các trình duyệt hiện đại.

* **Hỗ trợ dịch vụ AWS**

Thiết kế mặc định của AWS SDK dành cho JavaScript trên trình duyệt hỗ trợ toàn bộ các dịch vụ AWS tương thích CORS.

1. Demo:

* Cài đặt cấu hình cho DynamoDB bằng thư viện AWS và khởi tạo các biến truy cập đến DataBase:



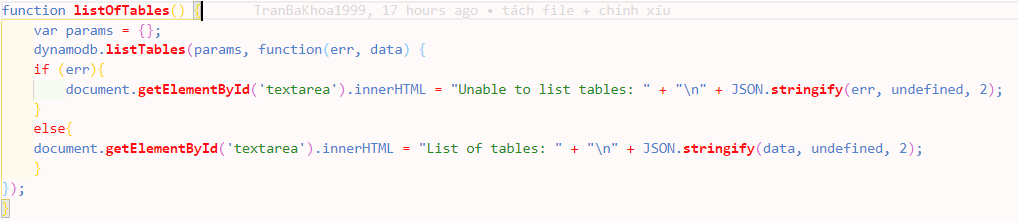
* Hàm tạo bảng:



* Hàm xóa bảng:



* Hàm hiển thị các bảng có trong DataBase:



* Hàm thêm một mục dữ liệu vào bảng:



* Hàm hiển thị tất cả thông tin của một mục được chọn trong bảng:



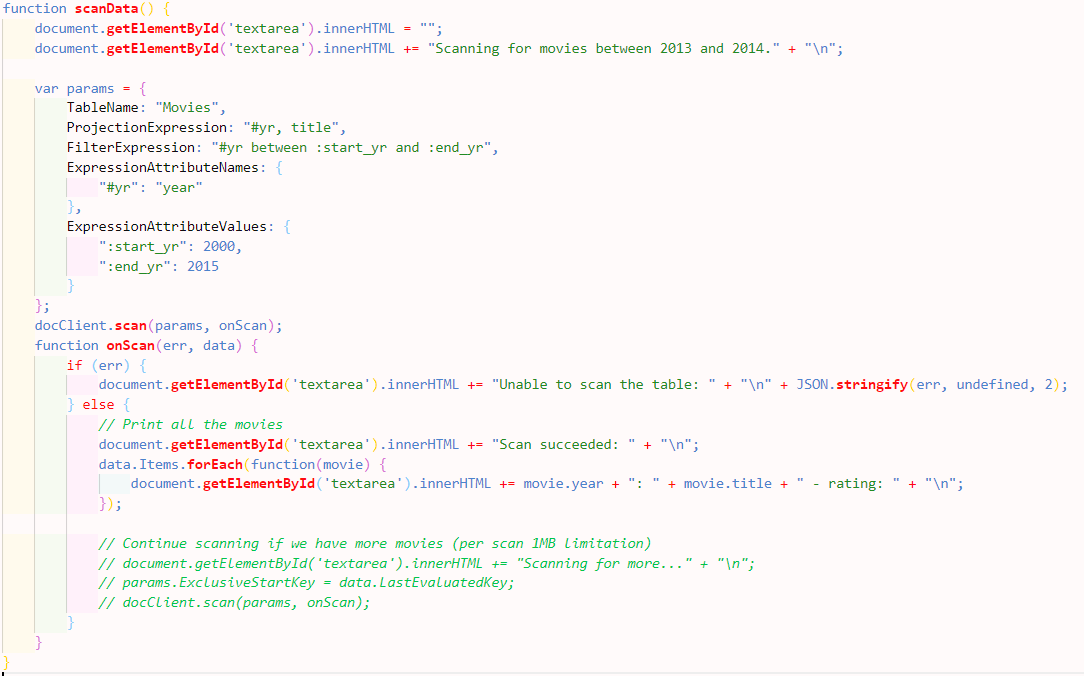
* Hàm xóa mục chỉ định:



* Hàm cập nhật mục chỉ định:



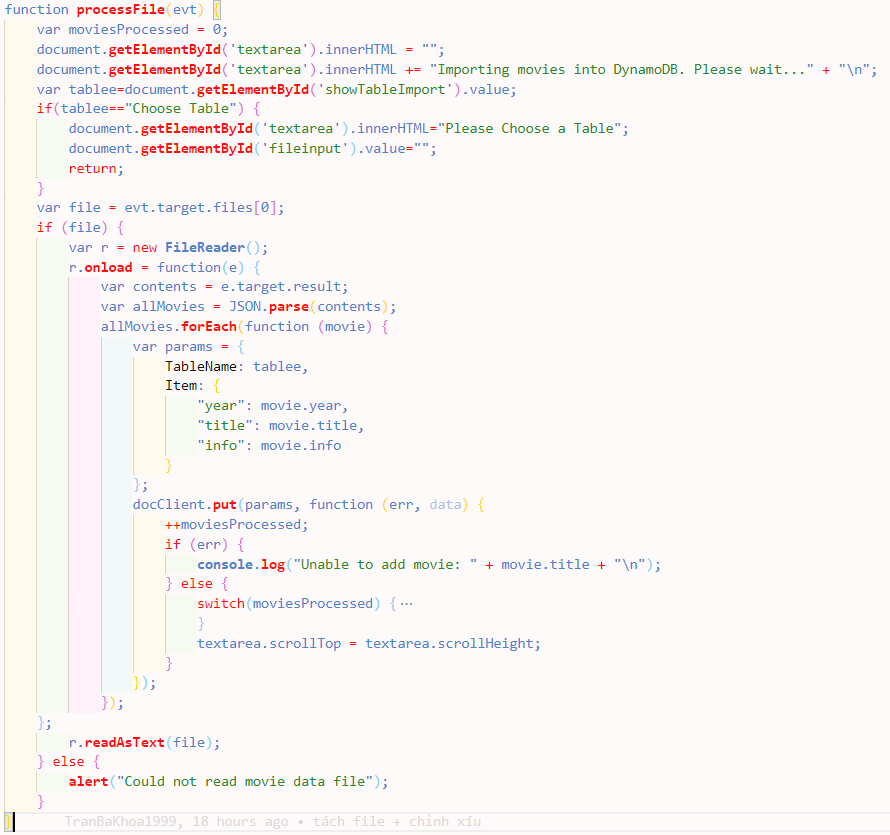
* Hàm quét dữ liệu theo điều kiện:



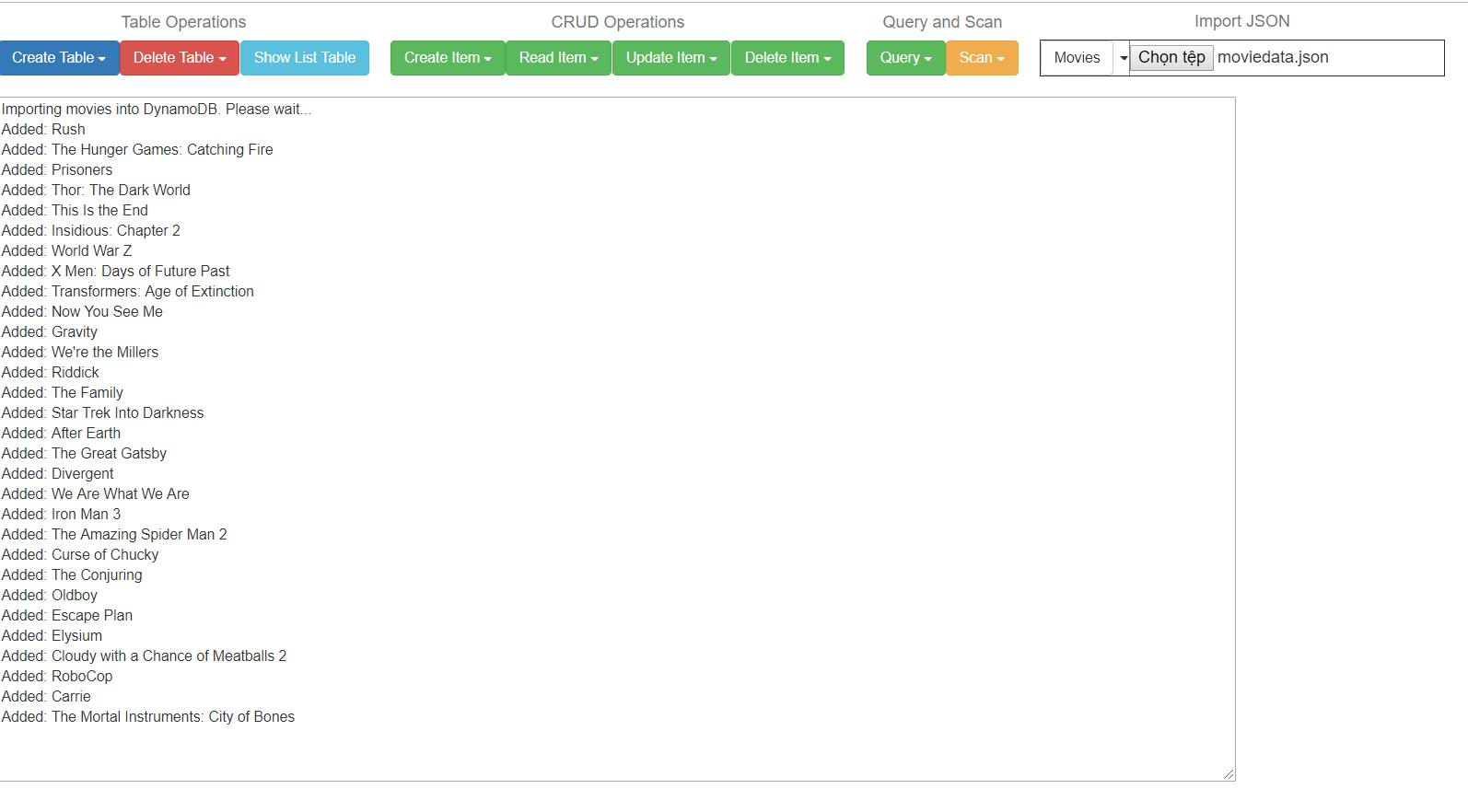
* Hàm truy vấn dữ liệu theo điều kiện:



* Hàm Nhập dữ liệu vào bảng từ file JSON:



* **Màn hình Web Demo:**



VI/ Tài liệu tham khảo

[https://aws.amazon.com/vi/dynamodb/](https://aws.amazon.com/vi/dynamodb/?fbclid=IwAR14RiDWiEy_xEzSPMxxrwY_rcZnTT8iZHq0UJPu1zeJKJSEBy_E3qWTC6E)

[https://viblo.asia/p/gioi-thieu-ve-dynamodb-phan-1-jamoG87nMz8P](https://viblo.asia/p/gioi-thieu-ve-dynamodb-phan-1-jamoG87nMz8P?fbclid=IwAR2Qh603ZpAQ3qK93AocvOOYxf2-XgVcHyvxbBPqPjRZWuYf1_50CXHwXdc)

[https://docs.aws.amazon.com/index.html?nc2=h\_ql\_doc\_do\_v](https://docs.aws.amazon.com/index.html?nc2=h_ql_doc_do_v&fbclid=IwAR1SH4_4ASZb9r3fu95N9aF1wwNe78bZSxr11vETFXnjWeWdP7iokN7xZeo)

[https://techblog.vn/gioi-thieu-ve-dynamodb-phan-1](https://techblog.vn/gioi-thieu-ve-dynamodb-phan-1?fbclid=IwAR329tlf7rZmridyoKP0MDr8GzC601tq3x1a8h5hBXFntcDo7N-vHQPGPkI)

[https://vnarchitect.wordpress.com/2016/03/22/dynamodb-co-gi-dac-biet/](https://vnarchitect.wordpress.com/2016/03/22/dynamodb-co-gi-dac-biet/?fbclid=IwAR2F6YRyO4rDSJSXzWzzzr4hb4vGhupTuVwnz1EQ2kNwOsK6ogyisCNwf78)

[https://toidicodedao.com/2015/09/24/nosql-co-gi-hay-ho-tong-quan-ve-nosql-phan-1/](https://toidicodedao.com/2015/09/24/nosql-co-gi-hay-ho-tong-quan-ve-nosql-phan-1/?fbclid=IwAR2Qh603ZpAQ3qK93AocvOOYxf2-XgVcHyvxbBPqPjRZWuYf1_50CXHwXdc)

[https://techmaster.vn/posts/35127/nhung-diem-khac-biet-giua-sql-va-nosql](https://techmaster.vn/posts/35127/nhung-diem-khac-biet-giua-sql-va-nosql?fbclid=IwAR2bt6IeQ7Khv5VNWXtOoVEBP5VoLAsNgwMVx5CHmHJl4Ux6E0N9gfkO9H0)